

(DATA-07-1-1001) - THE POWER INDUCED EFFECTS
MODULE: A FORTRAN CODE WHICH ESTIMATES LIFT
INCREMENTS DUE TO POWER INDUCED EFFECTS FOR
V/STOL FLIGHT. Report, 1 Jan. 1990 - 30 Sep.
1991 (California Polytechnic State Univ.)

101-30000

Unclass

62/02 0037936

**THE POWER INDUCED EFFECTS MODULE:
A FORTRAN CODE WHICH ESTIMATES LIFT INCREMENTS DUE TO
POWER INDUCED EFFECTS FOR V/STOL FLIGHT**

GRANTOR - NASA AMES RESEARCH CENTER

GRANTEE - CAL POLY STATE UNIVERSITY

GRANT NUMBER NCC 2-664

1/1/90 - 9/30/91

Principal Investigator

Dr. Doral R. Sandlin

Student Investigator

Kipp E. Howard

Cal Poly State University
San Luis Obispo, CA 93407

July 1991

ABSTRACT

THE POWER INDUCED EFFECTS MODULE:

A FORTRAN CODE WHICH ESTIMATES LIFT INCREMENTS DUE TO

POWER INDUCED EFFECTS FOR V/STOL FLIGHT

Kipp E. Howard

The objective of this project was to produce a user friendly FORTRAN code which can be used for preliminary design of V/STOL aircraft. It estimates lift increments, due to power induced effects, encountered by aircraft in V/STOL flight. These lift increments are calculated using empirical relations developed from wind tunnel tests and are due to suckdown, fountain, ground vortex, jet wake, and the reaction control system. The code can be used as a preliminary design tool along with NASA Ames' Aircraft Synthesis preliminary design code or as a stand-alone program for V/STOL aircraft designers.

The Power Induce Effects Module was validated using experimental data supplied by NASA-Ames Research Center, McDonnell Aircraft Company and data computed from lift increment routines developed by Richard E. Kuhn. Results are presented for many flat plate models along with the McDonnell Aircraft Company's MFVT V/STOL preliminary design and a 15% scale model of the YAV-8B, "Harrier", V/STOL aircraft.

It was found the Power Induced Effects Module predicts trends and magnitudes of lift increments verses aircraft height above the ground, well. The code was also found to predict only the magnitudes of lift increments verses aircraft forward velocity, well. More experimental results are needed to determine how well the code predicts lift increments as they vary with jet deflection angle and angle of attack.

TABLE OF CONTENTS

	<u>Page</u>
LIST OF FIGURES	vi
NOMENCLATURE	viii
 CHAPTER 1	
Introduction	1
 CHAPTER 2	
Lift Increment Descriptions	3
Suckdown.....	3
Fountain Lift	4
Fountain Arms.....	5
Fountain Core.....	7
Lift Improvement Devices.....	7
Ground Vortex.....	7
Jet Wake.....	8
Reaction Control System.....	10
Trends and Comparisons of the Lift Increments.....	12
 CHAPTER 3	
Computational Development	13
Variables.....	13
Independent Variables	13
Configuration Variables	13
Control and Result Variables.....	14
Hover	14
Suckdown.....	15
Fountain Lift	16
Lift Improvement Devices.....	18
Forward Flight / Jet Deflection Angle / Angle of Attack.....	19
Suckdown/Fountain.....	19
Ground Vortex.....	20
Jet Wake.....	21
Reaction Control System.....	22
 CHAPTER 4	
Computer Code Description.....	24
Philosophy	24
Control.....	24
Input	26
Preliminary Calculations.....	26

Lift Increments Routines	27
Height.....	27
Forward Velocity.....	28
Height, Forward Velocity, and Jet Deflection Angle.....	28
Height, Forward Velocity, Jet Deflection Angle, and Angle of Attack	29
Output.....	29
Limitations	30
CHAPTER 5	
Results and Discussion/Validation	31
Configurations.....	32
Flat Plate Models.....	33
Mixed-Flow Vectored-Thrust Configuration.....	33
YAV-8B	33
Hover	34
Disk	35
Body/Wing-Body	35
Delta.....	38
Mixed Flow Vectored Thrust	39
YAV-8B	40
Forward Flight.....	41
MFVT.....	42
YAV-8B	45
CHAPTER 6	
Conclusions and Recommendations	47
BIBLIOGRAPHY.....	49
APPENDIX A	
PIE FORTRAN Code.....	51
Coppie.....	51
Inpie.....	64
Planmo.....	74
Finvar.....	82
Sdarea.....	104
Integra	130
Diabar.....	130
Hcall	140
Hov_ge	145
Sfcall.....	155
Rescal.....	162
Resind	164
Gvcall.....	167
Stolgv.....	171
Wkcall	173
Jiepie.....	183
Stolwk.....	188
Piep	190
Dist.....	210
Perpdis.....	211
Linterp.....	211
Crossin	211

Linecro.....	212
Polyar.....	213
Centar.....	214
Ssen.....	217
Jietab.....	220

APPENDIX B

Power Induced Effects Module User's/Programmer's Manuals.....	223
---	-----

LIST OF FIGURES

	<u>Page</u>
Figure #1. Suckdown a) Out-of Ground Effect and b) In-Ground Effect	4
Figure #2. Photograph of a jet in a water tunnel illustrating the entrainment of air from the calm surroundings around the edge of the plate. (Reference 11)	4
Figure #3. Photograph of the flow field between two jets with the same nozzle pressure (Reference 11).	5
Figure #4. Planform view of jet pattern with four jets and the flow from each of the jets	6
Figure #5. "Fountain Core and Arms" generated by a configuration with three or more jets - Side view of jet pattern (Reference 6).	6
Figure #6. Flow field under an aircraft in hover with lift improvement devices	7
Figure #7. Ground vortex created from the interaction between the freestream and the jets of a V/STOL aircraft near the ground	8
Figure #8. Photograph of the cross section of the wake six nozzle diameters downstream of a jet exiting perpendicular to the free-stream in a water tunnel. (Reference 11)	9
Figure #9. Pressure-coefficient contours on a flat plate with a perpendicular exiting jet. (Reference 14)	10
Figure #10. Roll control reaction control system nozzles and the flow field around the wing	11
Figure #11. V/STOL induced lift increments (Reference 8)	11
Figure #12. Fountain variables from a configuration with three jets	17
Figure #13. Typical variation of fountain lift increment using h' Method	18
Figure #14. Lift Improvement Device definitions from a triangular configuration with three jets.	19
Figure #15. Flow chart of the control program COPPIE. (Note - Program Flow is downward unless specified.)	25

Figure #16.	Half-planforms of configurations used for verification.....	32
Figure #17.	Comparison of calculated and measured total lift increments in hover for a 20-inch circular disk (Reference 15).....	35
Figure #18.	Comparison of calculated and measured lift increments in hover for the Body planform (Reference 15)	36
Figure #19.	Comparison of calculated and measured lift increments in hover for the Wing-Body planform (Reference 15).....	37
Figure #20.	Comparison of calculated and measured lift increments in hover for the Delta planform (Reference 15).....	39
Figure #21.	Comparison of calculated lift increments in hover between Kuhn's program and PIE for the MFVT, triangular 3-jet configuration (Reference 7).....	40
Figure #22.	Comparison of calculated and measured total lift increments in hover for 15% scale model of the YAV-8B (Reference 9).....	41
Figure #23.	Comparison of calculated lift increments in forward flight at an altitude of 6 ft. for the MFVT, lateral 2-jet configuration from Kuhn's program and PIE (Reference 7).....	42
Figure #24	Comparison of calculated lift increments in forward flight at an altitude of 6 ft. for the MFVT, triangular 3-jet configuration from Kuhn's program and PIE (Reference 7).....	44
Figure #25.	Comparison of calculated and measured total lift increments in forward flight for a 15% scale model of the YAV-8B (Reference 9).....	46

NOMENCLATURE

A	Total jet exit area
ACSYNT	NASA-Ames Research Center's Aircraft Synthesis preliminary design code
AOA	Angle of attack of the aircraft with respect to the freestream
COPPIE	Control program for PIE
C _p	Coefficient of pressure
ΔC _p	Constant in expression for the contribution of the ground vortex
d	Diameter of single nozzle
\bar{D}	Angular mean diameter of the planform $\bar{D} = \frac{1}{\pi} \int_0^{2\pi} r d\theta$
d _e	Effective jet exit diameter (the diameter of a single jet which would have the same exit area as the sum of all the jets.)
DFL	Jet deflection angle with respect to the fuselage
e	Half the distance between adjacent jets (See Figure #12)
Exp	Experimental results
FORTTRAN	Formula translation - computer programming language
GE	In-ground effect
h	Height of aircraft above the ground (See Figure #12)
Δh	wing height above nozzles of configuration
h'	Critical height for h' Method (See Figure #13)
h ₁	Height at which the rate of change of lift with height changes
h ₂	Maximum height at which the ground vortex effects are felt
h _f	Height to which fountain effects are felt
H _t	Height of aircraft above the ground
K'	Constant in h' Method
K _A	Constant in expression for the contribution of the fountain arms
K _b	Adjustment factor for the proximity of the RCS jet to the wing tip
K _c	Adjustment factor for the proximity of the RCS jet to the wing trailing edge
K _C	Constant in expression for the contribution of the fountain core

K_h	Wall jet position factor
K_L	Percentage of lift which LIDs increase over configurations without LIDs
K_S	Constant in expression for multiple jet suckdown (correction factor)
l	Length of configuration
ΔL	Absolute change in total lift as the result of a power induced effect
LID	Lift improvement device
MCAIR	McDonnell Aircraft Company
MFVT	Mixed-flow vectored-thrust
N	Number of fountain arms
NASA	National Aeronautics and Space Administration
OGE	Out-of-ground effect
P	Pressure
PIE	Power Induced Effects Module
P_n/P	Nozzle pressure ratio
RCS	Reaction control system
S	Total planform area
S'	Actual surface area between the jets (See Figure #12)
S''	Potential surface area between the jets (See Figure #12)
sf	Suckdown/Fountain
STOVL	Short takeoff and vertical landing
T	Thrust
V	Velocity
V/STOL	Vertical/short takeoff and landing
V_e	Ratio of the aircraft dynamic pressure to the jet dynamic pressure
V_o	Forward velocity of aircraft
w	Width of configuration
x	Distance of RCS jet ahead of wing trailing edge
y	Spanwise distance of RCS jet from the wing tip (RCS section only)
y	Spanwise extent of the jet on the planform (See Figure #12)
Y	Maximum spanwise extent of jet on planform (See Figure #12)
YAV-8B	Harrier, a V/STOL jet fighter aircraft (Aircraft shown in Figure #7)

Greek

δ	Jet deflection angle
θ	Half the angle between adjacent jets and the center of the jet pattern

λ'	Exponent in h' Method
λ_A	Exponent in expression for contribution of fountain arms
λ_C	Exponent in expression for contribution of fountain core
λ_S	Exponent in expression for multiple jet suckdown
Σ	Summation

Subscripts

A	Fountain arm
B or b	Body
C	Fountain core
F	Fountain
f	Front jets
hover	Results from hover calculations
hw	High wing
j	Jets
JW or wake	Jet Wake
L	LIDs
O	Unaffected by pressure ratio
OGE	Out-of-ground effect
P	Affected by Pressure ratio
r	Rear jets
RCS	Reaction Control System
S	Suckdown
sf	Suckdown/Fountain
square	Square planform
V	Ground Vortex
W	Wing
wb	Wingbody
x	Individual fountain arm of multiple fountain arm configuration
∞	Conditions at infinity or of the freestream

CHAPTER 1

Introduction

Preliminary design of modern aircraft is labor intensive, time consuming and requires detailed knowledge of several engineering disciplines. Interactive computer programs have been developed which reduce the time necessary to perform the calculations involved and permit the engineer to quickly size an aircraft and determine if the design can meet specifications. The Aircraft Synthesis (ACSYNT) code developed by NASA is one of the existing preliminary design and analysis codes. This code will assist an aircraft designer in sizing and design of conventional takeoff and landing aircraft. An effort is now being made to update the code to include sizing and design of vertical and short takeoff and landing (V/STOL) aircraft .

V/STOL aircraft experience changes in the overall lift due to power induced effects. Even though the magnitude of these changes in lift, or lift increments, might be small compared to the installed thrust of an aircraft, they must be estimated if accurate predictions of aircraft performance are to be made. For example, an error of three percent in the total lifting capability in hover is about a 10% error in the prediction of the aircraft range (Reference 3).

In the past, lift increments for simple configurations were predicted by using wind tunnel data to generate empirical models. As more configurations were tested, the empirical models became increasingly more complicated to the point where hand calculations have become impractical and tedious. Computer programs have been developed to calculate each lift increment individually. These programs have proved adequate although the user needs to be knowledgeable about the lift increments, to make many pre-calculations for the

inputs, and to combine the results in order to find the total lift increment. A need exists for a program which does not require detailed knowledge of lift increments, requires only a minimum amount of input from the user and can output individual lift increments and/or total lift increments in any form desired by the user.

The objective of this project was to produce a user friendly FORTRAN code that estimates the lift increments, due to power induced effects, encountered by aircraft in V/STOL flight using the method developed by Kuhn and Stewart (References 6 and 13). The code was developed to enable the programmer to easily include moment increment calculations and customized output. The method uses empirically derived relations based on aircraft configuration and flight regime to calculate the lift increments. The code, which is titled the Power Induced Effect Module (PIE), calculates lift increments due to aerodynamic forces that result from the interaction of the lifting jets with the surrounding environment. These lift increments are due to the suckdown, fountain, ground vortex, jet wake, and the reaction control system. PIE is a stand-alone application, written in FORTRAN, which creates lookup tables that are compatible with ACSYNT. PIE can be easily incorporated into ACSYNT at a later date and made transparent to the user.

CHAPTER 2

Lift Increment Descriptions

Descriptions of the lift increments are presented to assist the reader to better understand this report. Lift increments are considered to be changes in the lift of an aircraft. This project accounts for lift increments which result from the operation of the power plant of a V/STOL aircraft and the interaction between its lifting jets, surrounding air mass, airframe, and landing surface. These increments either add or subtract from the total lift of the aircraft during hover or transition. They are a result of the suckdown, fountain, ground vortex, jet wake and reaction control system suckdown effects. The power induced lift increments which are not accounted for in this project are those dealing with turning and pressure losses in the lifting nozzles, recirculation, and hot-gas ingestion.

Suckdown

The jet exhaust which supports a V/STOL aircraft, entrains its surrounding air mass around the aircraft (Figure #1a). This entrainment is due to viscosity and it is greatest near the jet. The downward flow produced from this entrainment causes a downward force on the airframe (Reference 12) because the wings and body of the aircraft act as flat plates perpendicular to the flow.

Figure #2 is a photograph of a jet flow exiting normal to a flat plate. It can be seen that the entrainment of the flow causes the fluid particles to separate as they flow from above the plate around to the jet (Reference 11). This separated region causes negative pressures on the underside of the plate, sucking it down.

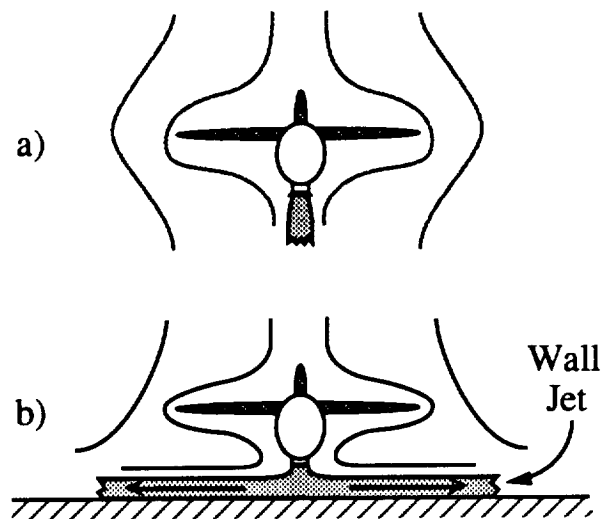


Figure #1. Suckdown a) Out-of Ground Effect and b) In-Ground Effect

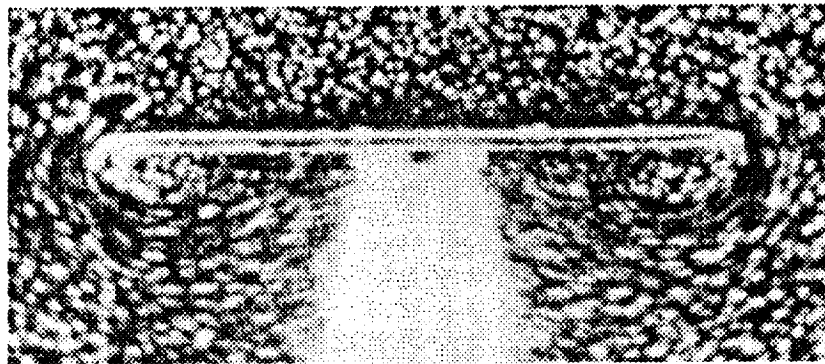


Figure #2. Photograph of a jet in a water tunnel illustrating the entrainment of air from the calm surroundings around the edge of the plate. (Reference 11)

Fountain Lift

When a single jet is near and perpendicular to the ground, a wall jet is radiated out in all directions from the point of impingement, similar to Figure #1b. When two or more jets are present and widely separated, their wall jets meet on the axis of symmetry between

the jets and since there is no other place to go, they merge and rise upwards, creating a fountain underneath the aircraft.

This effect can be seen in Figure #3 which is a side-view photograph of two nozzles exhausting perpendicular to the ground. The streamlines of the flow are shown by paint streaks on the ground plane and the vertical plane. The plane of symmetry and the actual fountain can be easily seen between the two nozzles.

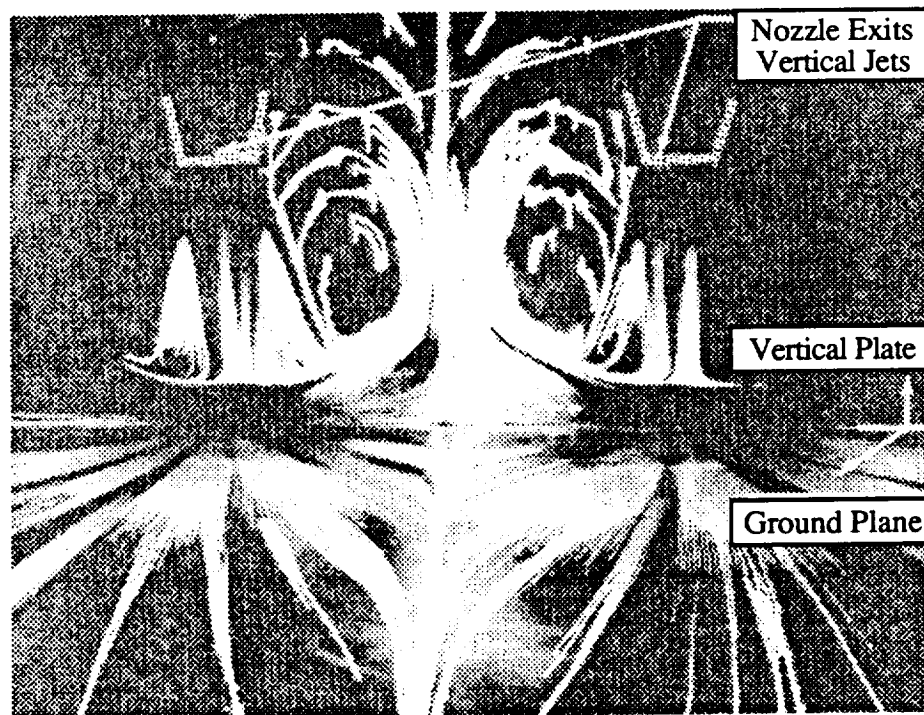


Figure #3. Photograph of the flow field between two jets with the same nozzle pressure (Reference 11).

Fountain Arms

The fountain arms are created when exhaust from any two neighboring jets meet on the ground at the plane of symmetry between the jets. This is shown between any two adjacent jets in Figure #4. When the two flows meet, they merge and rise upward in a relatively thin fan-shaped sheet as seen in Figure #5. This upward flowing sheet pushes

upward on the lower surface of the aircraft which gives the aircraft a positive lift increment. The upward flowing sheets exiting the vicinity of the jet pattern are referred to as the arms of the fountain or "fountain arms."

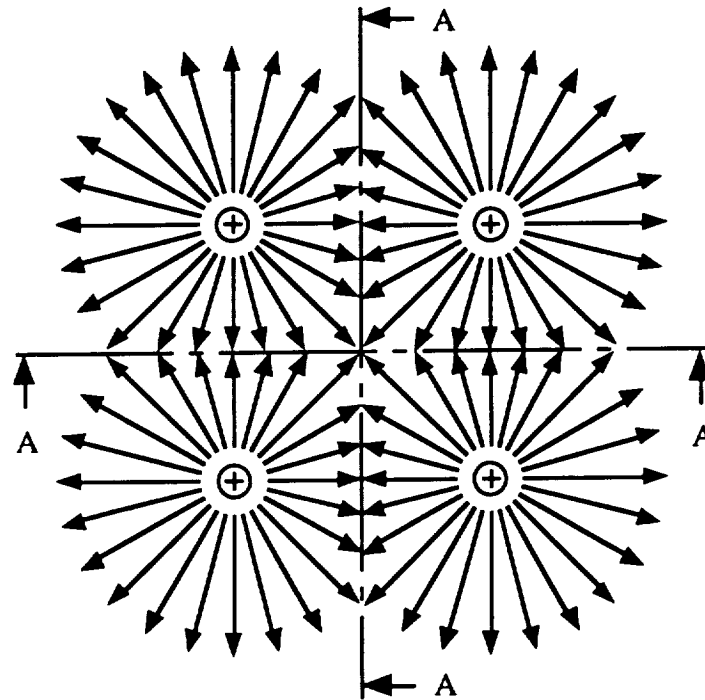


Figure #4. Planform view of jet pattern with four jets and the flow from each of the jets (Reference 6).

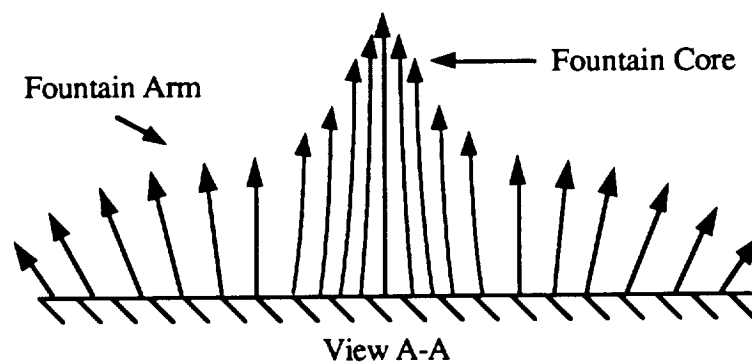


Figure #5. "Fountain Core and Arms" generated by a configuration with three or more jets - Side view of jet pattern (Reference 6).

Fountain Core

The fountain core is generated when the fountain arms created from three or more jets meet at the center of the jet pattern. This fountain core has higher upward velocities than those of the fountain arms (Reference 11) because multiple fountain arms converged at the center of the jet pattern as shown in Figure #4 and #5. These higher upward velocities increase the pressure underneath the aircraft imparting an upward force on its lower surface.

Lift Improvement Devices

The lift due to the fountain effect can be increased by the use of lift improvement devices or LIDs. These LIDs are strakes on the lower surface of the aircraft. The LIDs redirect the fluid downward which produces larger upward forces on the aircraft. The flow field underneath an aircraft with LIDs is shown in Figure #6.

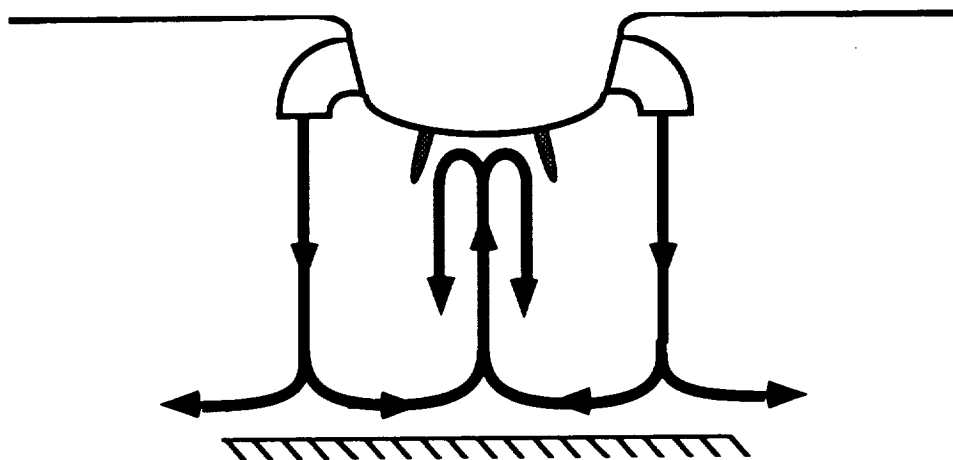


Figure #6. Flow field under an aircraft in hover with lift improvement devices

Ground Vortex

A ground vortex forms when the wall jet, from a vertical jet impinging on the ground, physically interacts with the freestream. The freestream causes the wall jet along the ground to roll up into a vortex which is swept in the direction of the freestream

(Reference 2). Figure #7 is a schematic of an aircraft operating close to the ground and shows the ground vortex due to the interaction between the wall jet and the crossflow.

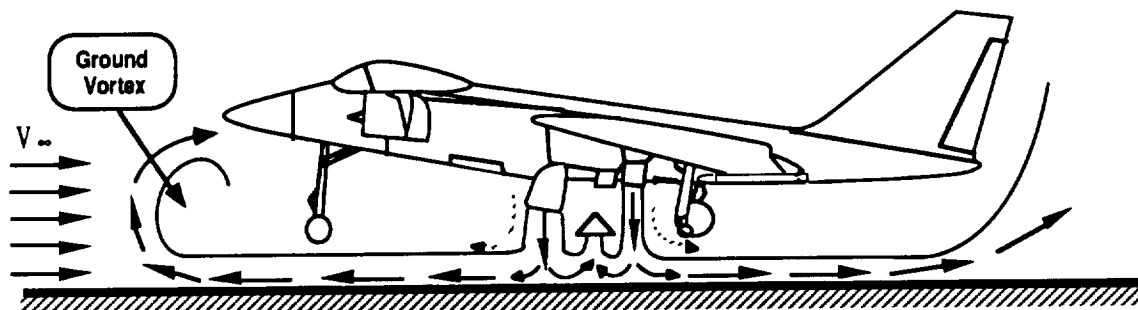


Figure #7. Ground vortex created from the interaction between the freestream and the jets of a V/STOL aircraft near the ground

The ground vortex contributes to the lift increment by causing increased flow underneath and near the aircraft. This flow reduces the pressure on the underside of the wing and body which in turn reduces the lift on the aircraft. The magnitude of the lift increment caused by the ground vortex is dependant on ground proximity and aircraft speed. If the aircraft is high enough or fast enough, the ground vortex is swept completely under the airframe causing no decrease in lift. As the aircraft approaches the ground and/or slows down, the loss in lift increases as more of the vortex interacts directly with the airframe.

Jet Wake

A jet wake occurs when a jet exits perpendicular to the freestream. The jet is deflected in the direction of the freestream and is quickly transformed into a rolled up vortex pair. Initially, shedding of the vortex pairs are analogous to the vortices which are shed from a solid cylinder placed in a crossflow. Due to the curvature and mixing of the jet in the freestream, the jet is transformed into a vortex pair and their axes become parallel to the freestream. These vortex pairs can be seen in Figure #8, which is a photograph of a cross section of an actual jet exiting perpendicular to the freestream in a water tunnel. The

white portion is the jet flow and the white dots are air bubbles which are induced into the vortices surrounding the jet flow (Reference 11). The vortices created from the jet wake are the primary cause of the interference effects in transition flight. The vortices change the flow field near the airframe which induces additional suction pressures on the surfaces of the aircraft (Reference 10).

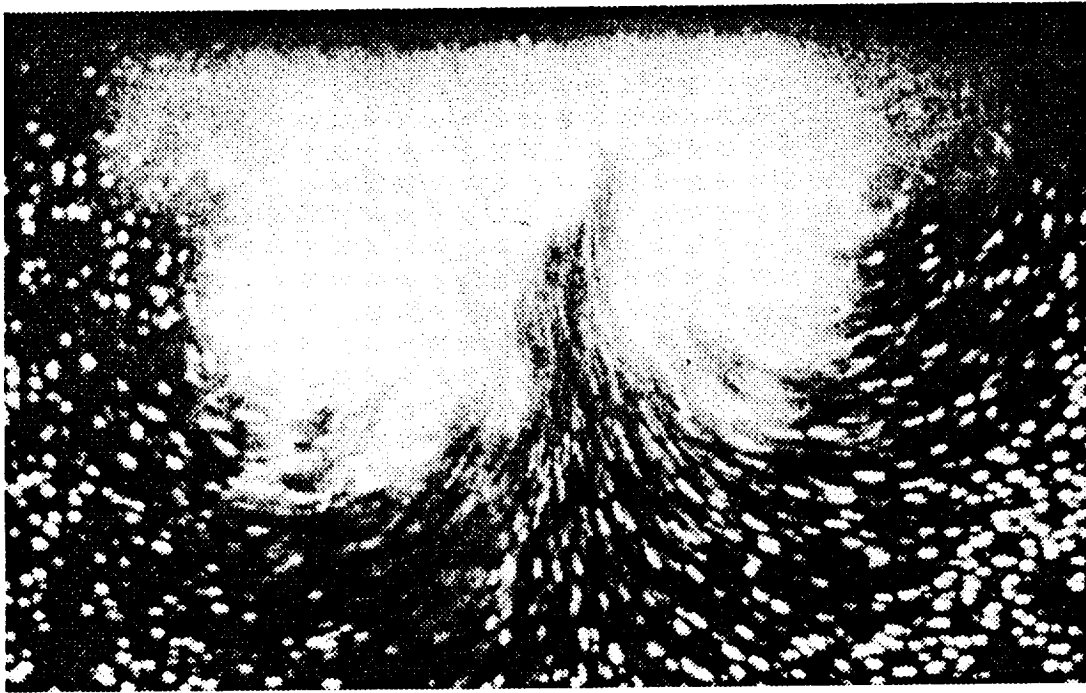


Figure #8. Photograph of the cross section of the wake six nozzle diameters downstream of a jet exiting perpendicular to the free-stream in a water tunnel. (Reference 11)

The negative lift increment created by the jet wake is due to large negative pressures developed behind the jet, as shown in Figure #9. This figure depicts pressure contours around a jet exiting perpendicular to the free-stream from a flat plate. It can be seen the jet produces a region of positive pressures upstream of the jet and a much larger region of stronger negative pressures laterally and downstream of the jet (Reference 14).

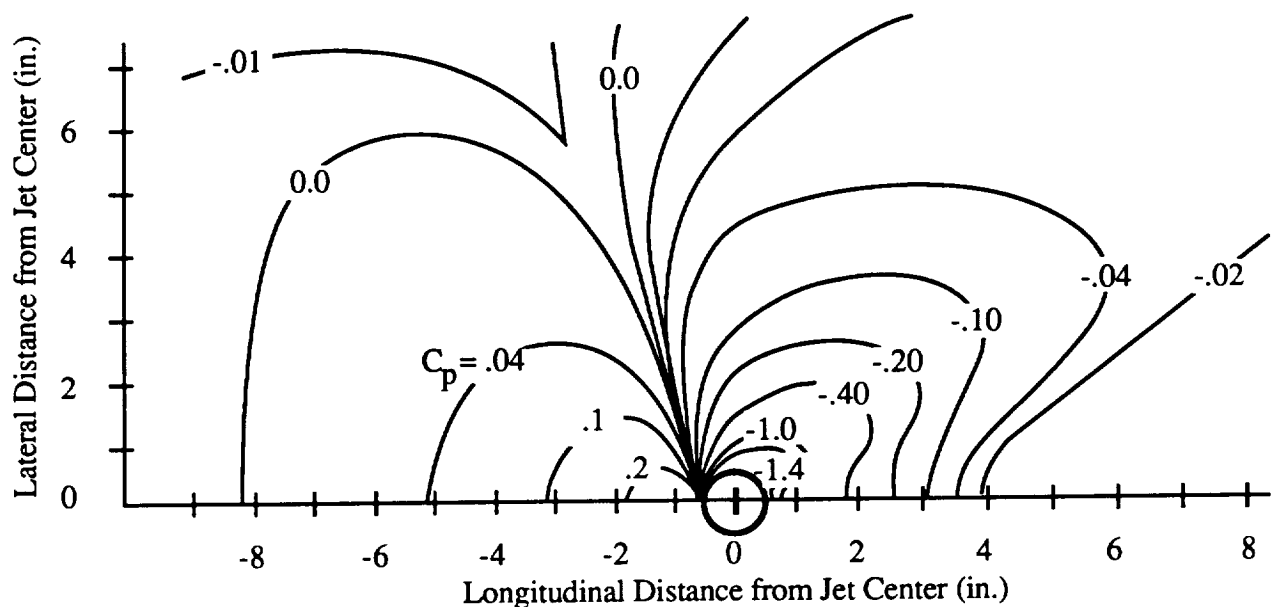


Figure #9. Pressure-coefficient contours on a flat plate with a perpendicular exiting jet. (Reference 14)

Reaction Control System

Vertical/short take-off and landing aircraft require pitch, roll, and yaw control when in hover or transition. Typically a portion of the exhaust is bled off from the engine and ducted to the extremities of the aircraft to the reaction control system (RCS). Nozzles on the wing tips are used for roll control and nozzles on the nose and tail can be used for pitch and yaw control. Studies have been performed on the nozzles used for roll control to determine their effectiveness in hover or forward flight. (This report/program will only account for the lift increments resulting from the roll control RCS nozzles due to the unavailability of data for pitch and yaw control nozzles.) It was found the roll control nozzles typically encounter the same type of suckdown and jet wake effects which the main lift engines encounter. These losses are due to the entrainment action of the jet, and the interaction of the free stream and the jet flow (jet wake losses). (See Figure #10.) Both of these effects induced suction pressures on the lower surface of the wing beside and behind

the jet causing a down load on the wing which reduced the effectiveness of the jet. (The magnitude of this lift increment is small compared to the total lift increment, but it is important when determining the maximum rolling moment which the RCS can produce. This will become much more important when moment increment routines are added to the code.)

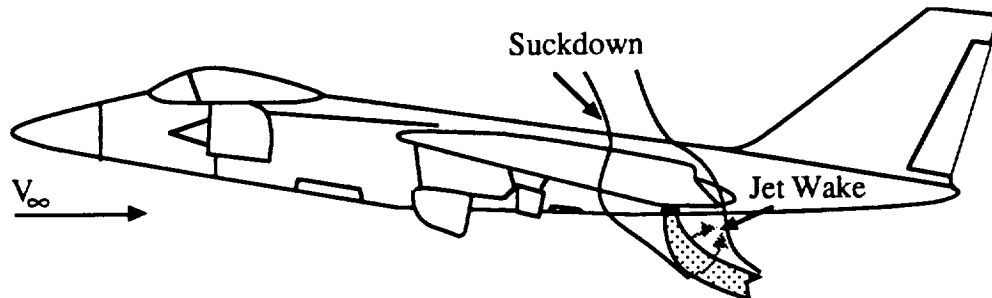


Figure #10. Roll control reaction control system nozzles and the flow field around the wing

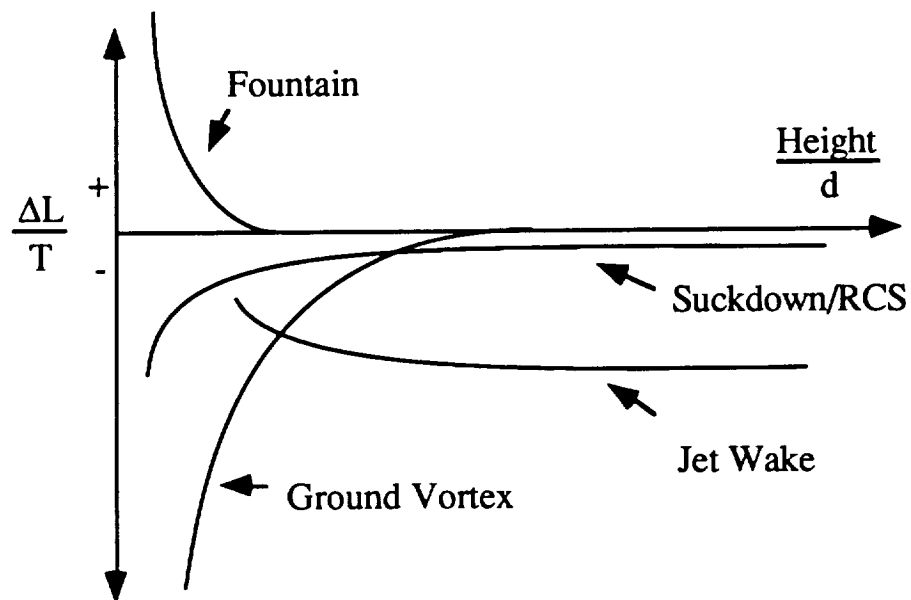


Figure #11. V/STOL induced lift increments (Reference 8)

Trends and Comparisons of the Lift Increments

Figure #11 is a graph that displays how lift increments vary with height above the ground. The normalized parameters are used so comparisons between different types and scales of configurations can be made. As can be noted from Figure #11, the ground has a profound effect on the lift increments. They vary near the ground and then level off to a constant out of ground effect (OGE) value as the altitude is increased.

CHAPTER 3

Computational Development

All relations developed for this project are empirical equations which were developed from wind tunnel data by Kuhn (Reference 6) and Stewart and Kuhn (Reference 13). These relations include many configuration and flight regime specific equations. The general equations will be presented with a mention of the specific changes to these equations for given configurations and flight regimes.

Variables

The variables which are present in the PIE routines can be divided into three categories: independent, configuration and control variables. All variables have the capability of being changed by the user. See Appendix B, the User's/Programmer's Manual, for more details.

Independent Variables

The independent variables are the flight conditions of the aircraft. These variables are aircraft height above the landing surface, forward velocity, nozzle deflection angle, and angle of attack. They were chosen as independent variables because they define the flight conditions of any particular aircraft design. Also ACSYNT requires lift increments based on these independent variables.

Configuration Variables

Configuration variables define the configuration of a particular aircraft. These variables can be further divided into three categories. The first category are those variables

which define the aircraft wing, body and wing/body. These include widths, lengths, heights, areas, and positions of aircraft components. The second category defines the number, positions, diameters, areas, configurations, and performance of the nozzles. The third category of variables define the positions, configuration, and affected areas of fountain arms and core.

Control and Result Variables

The control variables track and control execution, and record errors. These variables include the indices which allow quick and convenient access to results. The result variables store results for each lift increment for each change in the independent variables. These result variables are stored and retrieved using indexed notation ($u(i,j)$) where 'i' and 'j' can be any positive integer. Due to memory constraints on the computer systems, some of the results are stored in disk files instead of computer memory but are still stored and accessed using the same indexed techniques. Many of the results are stored as components of the total lift increments. Frequently, these components are the lift increments calculated separately for the body, wing, front nozzles, and rear nozzles. Later these components are combined using the principle of superposition.

Hover

The only independent variable used in the hover calculations is height. The other independent variables are set to zero velocity, 90° nozzle deflection angle, and 0° angle of attack.

The method used in this study was developed by Kuhn (Reference 6) and its major assumption is the total induced lift increment developed on a V/STOL aircraft can be expressed as:

$$\frac{\Delta L}{T} = \frac{\Delta L_{\infty}}{T} + \frac{\Delta L_S}{T} + \frac{\Delta L_F}{T} + \frac{\Delta L_L}{T} \quad (1)$$

Suckdown

$\frac{\Delta L_{\infty}}{T}$ is the out-of-ground effect (OGE) suckdown lift increment. The equation for the OGE suckdown is:

$$\frac{\Delta L_{\infty}}{T} = -.00022 \sqrt{\frac{S}{A}} \left[\left(\frac{P_n}{P_{\infty}} \right)^{-.64} \frac{\Sigma \pi d}{d_e} \right]^{1.58} \quad (2)$$

where S is the total planform area, A is the total jet exit area, P_n/P_{∞} is the nozzle pressure ratio, $\Sigma \pi d$ is the total jet perimeter, and d_e is the diameter of an equivalent single jet having area A . Experiments have shown that the OGE lift increment is reduced if the nozzles are extended below the surface of a body. The OGE lift increment for a high wing configuration is lower than the low wing configuration and the decrease is related to the square root of the wing height over nozzle diameter as shown in equation (3).

$$\left(\frac{\Delta L_{\infty}}{T} \right)_{hw} = \left(\frac{\Delta L_{\infty}}{T} \right)_b + \left[\left(\frac{\Delta L_{\infty}}{T} \right)_{wb} - \left(\frac{\Delta L_{\infty}}{T} \right)_b \right] \left[1 - .4 \sqrt{\frac{\Delta h}{d_e}} \right] \quad (3)$$

where Δh indicates the wing height above the nozzles of the configuration, and the subscripts hw indicates a high wing, b indicates the body, and wb indicates the wingbody.

$\frac{\Delta L_S}{T}$ is the additional lift increment due to suckdown experienced in-ground effect (GE). It was first determined experimentally for single jet configurations and then a correction factor was developed for multiple jet configurations. The equation for in-ground effect suckdown is:

$$\frac{\Delta L_S}{T} = K_S (-0.015) \left[\frac{\frac{h}{d_e}}{\bar{D} - 1.0} \right]^{2.2 - .24 \left(\frac{P_n}{P} - 1 \right)} \quad (4)$$

Where in addition to the variables mentioned above, \bar{D} is the angular mean diameter of the planform, and K_S is the multiple jet correction factor which is a function of configuration geometry. This GE suckdown also changes with wing height similar to the OGE

suckdown. This is because the wing does not experience the same magnitude of entrainment as the lower surface of the body.

Fountain Lift

Two methods were developed by Kuhn to calculate the fountain lift, one for widely spaced jets, the Basic Method, and one for closely spaced jets, the h' Method.

Basic Method. The Basic Method was developed for widely spaced jets with equal thrusts. $\frac{\Delta L_F}{T}$ is the overall fountain lift increment which is a combination of the lift increment developed by the fountain arms and fountain core.

$$\frac{\Delta L_F}{T} = \frac{\Delta L_A}{T} + \frac{\Delta L_C}{T} \quad (5)$$

Equations (6) and (7) are used to calculate the fountain arm contribution. Equation (6) is the lift increment for an individual fountain arm 'x'. Equation (7) combines the fountain arms in the configuration.

$$\frac{\Delta L_{Ax}}{T} = \frac{2(Y_x S_x')^{.835}}{N(e_x S_x'')} \left(\frac{e_x}{e_x + h} \right)^2 \frac{y_x}{\sqrt{y_x^2 + (e_x + h)^2}} \quad (6)$$

$$\frac{\Delta L_A}{T} = \frac{1}{2} \left(\frac{\Delta L_{A,1}}{T} + \frac{\Delta L_{A,2}}{T} + \dots + \frac{\Delta L_{A,N}}{T} \right) \left(.7 \sqrt{\frac{\frac{h}{d_e}}{\frac{D}{d_e} - 1}} \right) \quad (7)$$

y_x is the spanwise extent of the fountain on the planform, Y_x is the maximum spanwise extent of the fountain on the planform, e is half the distance between adjacent jets, h is the height of the lowest surface above the ground, S' is the actual surface area between the jets, S'' is the potential surface area between the jets, and N is the number of fountain arms. These variables can be seen in Figure #12.

$$\frac{\Delta L_F}{T} = K' \left(\frac{h}{d_e} \right)^{\lambda'} \quad (10)$$

$$\frac{\Delta L_F}{T} = 0.033 \left(\frac{Dw}{d_{el}} \right) \left(\frac{d_e}{h} \right) \quad (11)$$

where K' and λ' are parameters calculated based on jet spacing, diameter, number, surface area, etc..., w is width of the configuration, and l is the length of the configuration. The two curves do not intersect but are joined by a straight line which is tangent to the curve of equation (10) and is projected to $\frac{\Delta L_F}{T} = 0$ at a height defined as h' . This is shown in Figure #13.

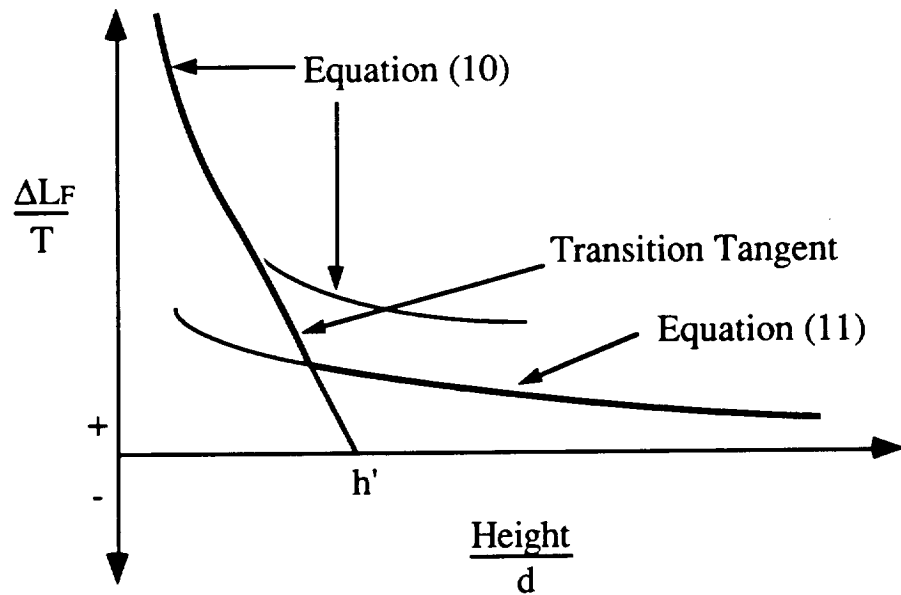


Figure #13. Typical variation of fountain lift increment using h' Method

Lift Improvement Devices

$\frac{\Delta L_L}{T}$ is the lift improvement devices (LIDs) lift increment. This lift increment is an addition to the total fountain by a percentage, K_L , of the total fountain lift. K_L can be determined using the area inclosed by the LIDs, ratio of perimeter by LIDs to the total

perimeter area enclosed by LIDs, area enclosed by jet centers, length-to-width ratio of jet pattern, and altitude of aircraft. These variables can be seen in Figure #14.

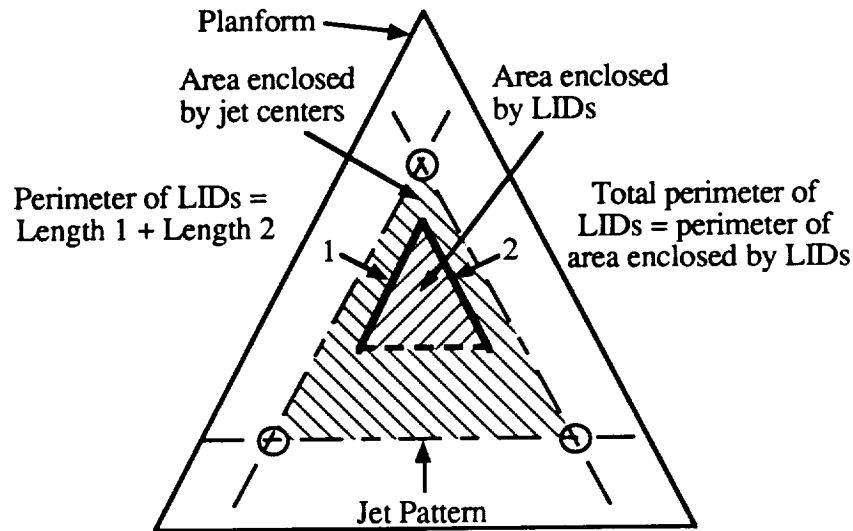


Figure #14. Lift Improvement Device definitions from a triangular configuration with three jets.

Forward Flight / Jet Deflection Angle / Angle of Attack

The calculations based on forward flight, jet deflection angle, and angle of attack use hover calculations as a basis for each lift increment calculation. The method's major assumption is the total lift can be expressed as:

$$\frac{\Delta L}{T} = \frac{\Delta L_S}{T} + \frac{\Delta L_F}{T} + \frac{\Delta L_{JW}}{T} + \frac{\Delta L_V}{T} \quad (12)$$

where the subscript S refers to the suckdown lift increment (no fountain), F refers to the fountain lift increment, JW refers to the jet wake lift increment, and V refers to the lift increment due to the ground vortex.

Suckdown/Fountain

The suckdown and fountain terms are calculated using similar methods. These methods calculate a factor based on the forward extent of the wall jet and the jet spacing of

the front nozzles. This factor, K_h , varies between one (the wall jet extending ahead of the configuration) and zero (the wall jet swept behind the configuration.) This factor is multiplied by the square of the sine of the jet deflection angle (δ). (Angle of attack is not used in these calculations.) This calculation produces a factor which modifies the suckdown and fountain terms for the effects of forward speed and jet deflection angle. The suckdown lift increment for forward flight and jet deflection angle is calculated as follows:

$$\frac{\Delta L_S}{T} = \left(\frac{\Delta L_S}{T} \right)_{\text{hover}} K_h \sin^2 \delta \quad (13)$$

The fountain lift increment is calculated using the fountain and LID lift increments from the hover calculations as shown in equation (14).

$$\frac{\Delta L_F}{T} = \left[\frac{\Delta L_F}{T} + \frac{\Delta L_L}{T} \right]_{\text{hover}} \sqrt{\frac{h_f}{h'}} K_h \sin^2 \delta \quad (14)$$

This lift increment calculation is very similar to equation (13) except for the $\sqrt{\frac{h_f}{h'}}$ term.

This term is included from the recommendation of Richard Kuhn. It is ratio of h' , which was described earlier and h_f , which is the maximum height to which the fountain effects are felt on the airframe.

Ground Vortex

The ground vortex term is calculated by assuming the configuration is composed of a body and a wing. The lift increment is based on the thrust, diameter and area of the front jets. There are two critical heights associated with the ground vortex lift increments; h_1 is the height at which the rate of change of lift with height changes and h_2 is the maximum height at which the ground vortex effects are felt. The equations are different for the wing and body and each of those are different for heights based on h_1 and h_2 . An example of these relations is the equation for a body at a height between h_1 and h_2 , equation (14).

$$\frac{\Delta L_v}{T} = \frac{T_f}{T} \cdot 18 \left(\frac{w}{l} \right)_f^{-.2} \left[\left(\frac{S_B}{A_f} \right) \left(\frac{w}{l} \right)_B \right]^{.62} \sqrt{\frac{1}{V_e}} \Delta C_p [1 + \sin(\delta - 90)]^2 \left(\frac{h}{d_f} \right)^{-3.2} \quad (14)$$

The other equations are similar to equation (14) in complexity and format. The subscript f refers to the front jets, and B refers to the body configuration. T is the thrust, w is the width, l is the length, S and A are area, V_e is the ratio of the aircraft dynamic pressure to the jet dynamic pressure, ΔC_p is a factor based on V_e and height, and δ is the jet deflection angle.

Jet Wake

The lift increment due to the jet wake is the sum of the increments due to each jet on the wing and body. This is shown in equation (15) where B refers to the body planform, W refers to the wing planform, f refers to the front jets and r refers to the rear jets.

$$\frac{\Delta L_{wake}}{T} = \left[\left(\frac{\Delta L_f}{T} \right) + \left(\frac{\Delta L_r}{T} \right) \right]_B + \left[\left(\frac{\Delta L_f}{T} \right) + \left(\frac{\Delta L_r}{T} \right) \right]_W \quad (15)$$

The lift increment for the wake is figured by first calculating the lift increment out-of-ground effect and then including this term in the overall lift increment in-ground effect. The OGE jet wake lift increment is calculated using equation (16), which is the equation for the lift increment on a square planform with the jet at the center, and multiplying it by adjustment factors for the planform aspect ratio, longitudinal, vertical, and lateral position of the jet, distance of the jet center from the side of the body, jet deflection angle, non-circular nozzles, jet pressure ratio, and jet flap.

$$\frac{\Delta L_{w,square}}{T} = \left[-3 V_e^2 \left(\sqrt{\frac{S}{A}} - 1 \right)^{.67} + 35 V_e^{5.5} \left(\sqrt{\frac{S}{A}} - 1 \right) \right] \quad (16)$$

The OGE lift increment is included in equation (17) which is the equation to calculate the jet wake lift increment for an individual jet on a body planform.

$$\frac{\Delta L_{w,x}}{T} = \frac{\Delta L_{w,OGE}}{T} \left[1 - .7 \sqrt{\left(\frac{S}{A} \right)_B \left(\frac{w}{l} \right)_B} \left(\frac{\delta}{90} \right)^{1.34} \left(\frac{w}{l} \right)_j^{-.35} \left(\frac{h}{d} \right)^{-2} \right] \quad (17)$$

Similar equations are used in calculations for the wing planform. The subscript j refers to the jets

If the jets are placed near the trailing edge of the wing, there is a positive lift increment instead of a negative. This is due to the jet flap effect. In this case a positive lift increment is calculated and replaces the OGE term an equation similar to equation (17).

Reaction Control System

The effectiveness of the roll control jets near the wing tip during transition depends on the proximity of the control jets to the wing tip and the wing trailing edge, and jet pressure ratio (P_j/P_∞). The empirical equations developed for this lift increment are based on an integration of the pressure distributions on the surfaces surrounding the nozzles.

The lift increments are assumed to be made up of two terms, one which is not affected by pressure ratios (O) and another from the effect of pressure ratios above a critical pressure ratio of 1.893 (P).

$$\frac{\Delta L_{RCS}}{T} = \left(\frac{\Delta L}{T} \right)_O + \left(\frac{\Delta L}{T} \right)_P \quad (18)$$

where:

$$\left(\frac{\Delta L}{T} \right)_O = (3V_e^3 - 2.4V_e^2) \sqrt{\frac{S}{A_j}} + .41V_e^{2.2} \left(\frac{S}{A_j} \right)^{.688} \quad (19)$$

$$\left(\frac{\Delta L}{T} \right)_P = -.017V_e \left(\frac{S}{A} \right)^{.42} \left(\frac{P_j}{P_\infty} - 1.893 \right)^{.75} \quad (20)$$

The variable S is the area of the wing, A_j is the jet exit area, P_j is the jet exit pressure, and P_∞ is the atmospheric pressure. These equations reproduce wind tunnel data with reasonable accuracy up to effective velocity ratios of $V_e = .1$, planform-to-jet area ratios up to 7000, and jet pressure ratios up to 45.

The magnitude of the lift increment decreases as the jet is moved closer to the wing tip or trailing edge. This is because the total area near the jet is decreased which reduces the effect of the negative pressures caused by the jet. Equation (21), (22), and (23) show the jet location adjustment factors and how they affect the total RCS lift increment.

$$K_b = .25 + .2 \left(\frac{y}{d_e} \right)^{.58} \quad (21)$$

$$K_c = .25 + .06 \frac{x}{d_e} \quad (22)$$

$$\frac{\Delta L_{RCS}}{T} = \left[\left(\frac{\Delta L}{T} \right)_O + \left(\frac{\Delta L}{T} \right)_P \right] K_b K_c \quad (23)$$

K_b is the adjustment factor for the proximity of the jet to the wing tip, y is the distance from the trailing edge to the jet, K_c is the adjustment factor for the proximity of the jet to the wing trailing edge, and x is the distance from the wing trailing edge to the jet.

This term is usually not included in the total lift increment because its magnitude is much smaller compared to the total lift increment and therefore need not be calculated for the general configuration. These routines were included in PIE to allow rolling moment increment routines, which use this lift increment, to be included at a later date.

CHAPTER 4

Computer Code Description

Philosophy

The Power Induced Effects Module is written in a modular structure which makes it easy to understand and modify. The code is divided into five fundamental sections; control, input, pre-calculations, lift increments, and output. Each of these modules are divided into individual parts to make logic and coding easier to understand. (See Appendix A for a listing of all routines used in PIE.)

Control

The Power Induced Effects (PIE) module is controlled by the control program, COPPIE, which coordinates the execution of PIE. COPPIE's first task is to make calls to input and pre-calculation subroutines. These calls are made first so all variables are either defined by the user, assigned a default value, or calculated. Within these variables the number, limits, and steps are defined for each independent variable; height, velocity, nozzle deflection angle, and angle of attack. COPPIE steps from one to the maximum number of variations for each independent variable (i.e. If 15 heights are to be considered then COPPIE steps from one to 15 for height). The actual value for the independent variable is calculated and then the control routine for a lift increment(s) is called and/or the next independent variable is incremented. After all independent variables have been incremented up to their maximum values then the output subroutine is called. Figure #15 is a flow chart which shows the basic structure of COPPIE.

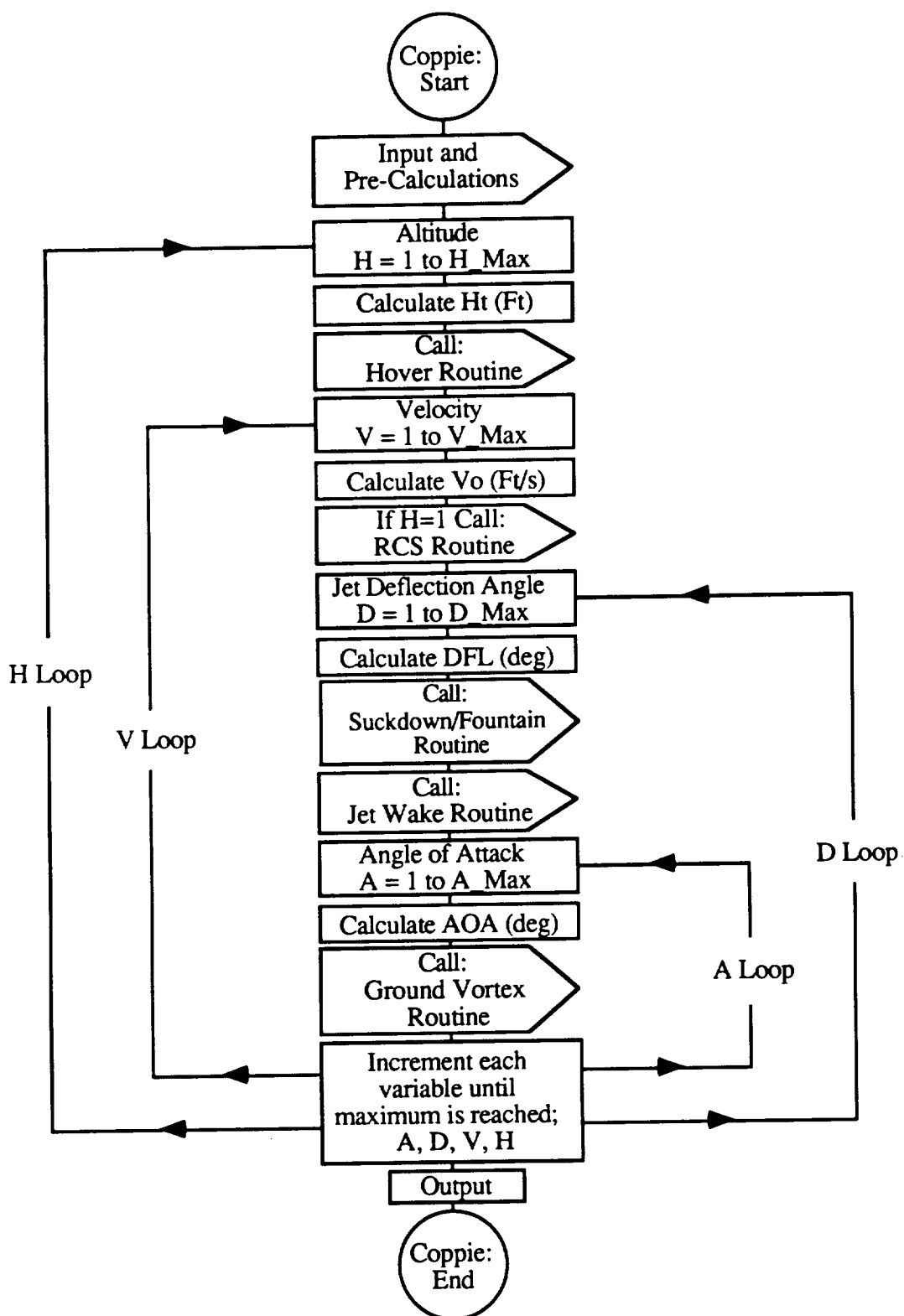


Figure #15. Flow chart of the control program COPPIE. (Note - Program Flow is downward unless specified.)

Input

Input routines for PIE perform two functions: input of variables, and check/modification of variables for consistency. The first routine of the input routines reads two input files. The first file contains all user defined variables. The second file contains X and Y coordinates for half the configuration planform. This routine first sets default values for all variables and then reads user defined variables from the first file. The advantage of this style of input over the input in the original programs (variables are entered in the code, hard-coded) is it allows the user to input as many or as few variables (above the minimum configuration variables) as the user desires. It also allows the user to save the configuration in a small, easy to manipulate files and rerun particular configurations as desired. The planform file allows a majority of calculation-intensive variables to be calculated by PIE, which alleviates the user from making tedious calculations.

The second routine assures planforms defined in the second file are consistent with the coordinate system defined in PIE. If the nose of the configuration does not lie on the origin, the routines modify all coordinates of the planforms and necessary configuration variables so the nose does lie on the origin. This makes sure PIE is able to interpret user inputs correctly.

Preliminary Calculations

The original programs have a few serious problems which make them tedious and difficult to use. First, these programs need extensive input in order to calculate the lift increment. These inputs include calculations for actual and potential surface areas between jets, and angular mean diameter of the planform which are tedious calculations because they require an integration over portions of the planform. There are also a number of less tedious calculations, but the number of them make the calculations time consuming for the user. Second, the user needs to know exactly how calculations are made so they can adjust variables so the results are computed correctly. For example, when the configuration has a high wing, areas for the fountain arms are smaller compared to configurations with low

wings, critical angular mean diameters are different, and some lengths based on the nozzles change. These procedures require detailed knowledge of lift increments and the overall method.

A desirable feature of the Power Induced Effects Module (PIE) is it contains a complete set of variables which defines the configuration. This allows lift increment control routines to pick and choose variables needed by lift increment routines. Many pre-calculations need to be made to obtain this complete set of variables. These pre-calculations take a number of hours by hand for each configuration and they change with each new configuration. This means if a new configuration needs to be completed, the calculations need to be refigured, which a computer can do quickly

Lift Increments Routines

The lift increment routines were divided into four sections. The sections are based on which independent variables are used to calculate the lift increment. The first section is lift increments calculated for hover. These lift increments vary with height. The second section contains lift increments which vary with velocity. The third section contains lift increments which vary with height, velocity, and jet deflection angle. The fourth section contains lift increments which change with height, velocity, jet deflection angle, and angle of attack.

Height

The hover routines calculate initial lift increments for the suckdown and fountain which vary with height. The RCS, ground vortex, and jet wake lift increments occur when a freestream is present so they are not included in these calculations. The out-of-ground effect (OGE) suckdown is calculated first and then corrections are made for in-ground effect (GE) based on the altitude of the configuration. The fountain terms are calculated based on their proximity to the ground using the "Basic" and "h" methods outlined by Kuhn (Reference 6).

The specific function of the hover control routine is to recognize when a high wing is present and make appropriate adjustment to variables which it sends to the lift increment routine.

Forward Velocity

The reaction control system (RCS) lift increment varies with velocity. Ground effects can be neglected since the size of the RCS jets are much smaller than their height above the ground. Corrections for jet placement are calculated first. The lift increment based on pressure ratio is calculated next and adjustment factors are applied to the results.

The specific function of the RCS control routine is to either execute the lift increment routine or set its result to zero. This is controlled with a flag variable which can be changed by the user.

Height, Forward Velocity, and Jet Deflection Angle

Suckdown, fountain, and jet wake lift increments vary with height, forward velocity and jet deflection angle. Suckdown and fountain terms are calculated inside the same increment routine because they both change similarly with the independent variables. A height factor is calculated based on the forward extent of the wall jet. Then the calculations use hover values multiplied by height, forward flight and nozzle deflection angle factors. The control routine for suckdown and fountain routine track the positions of the nozzles and whether there is a high wing and passes the necessary information to the suckdown and fountain lift increment routine.

The jet wake term is calculated using methods described by Stewart (Reference 13). Since the methods makes separate calculations for different planforms and for front and rear jets, the control program is much more detailed than the others. The control program first determines which kind of planforms are to be used: wing and body or wing/body (one planform which is a combination of the wing and body). It checks if the nozzles are near the trailing edge of the wing, for the jet flap effect, and then calculates the

OGE and overall lift increment for the jet wake from front and rear nozzles. This process is repeated for the next planform. The OGE and overall lift increments are calculated in separate routines due to the complexity of each calculation. The calculations for the wing and body are stored separately to allow contributions of each planform to be observed.

Height, Forward Velocity, Jet Deflection Angle, and Angle of Attack

The ground vortex term is calculated based on height, forward velocity, nozzle deflection angle, and angle of attack. This is the only term which varies with angle of attack.

The control routine for the ground vortex term is similar to the jet wake control routine with respect to the planforms. The rear nozzles do not contribute to the ground vortex so only the front nozzles are used in the calculations. Lift increments for the wing and body are calculated and stored separately to allow contributions of each planform to be observed

Output

Storage of the results of PIE are in files and arrays which can be recalled with the use of an index notation. This is done so any type of lookup table can be created by the user/programmer. This method of storage allows the programmer to configure an output table in any format using simple index notation when requesting a value (i.e. if the user wants a lift increment for the eighth height, sixth velocity, third deflection angle and fifth angle of attack, it can be recalled by setting the appropriate index variables; i, j, k, and l to 8, 6, 3, and 5 respectively). This feature allows easy access to the lift increments which facilitates many different types of look-up table output.

Currently, PIE creates one three-dimensional look-up table for ACSYNT. This table contains multiple two-dimensional tables based on deflection angle with each individual table based on height and velocity. This table is generated by a single, short subroutine which demonstrates how many other types of tables can be generated.

Other output options are created for viewing purposes. These tables are more examples of the many tables which can be generated. Each lift increment as well as the total lift increments can be viewed based on two of the four independent variables. Currently, outputs are text files which can be read by a plotting routine, developed at NASA Ames for the Silicon Graphics IRIS workstations, and a Macintosh graphing application called KaleidaGraph™.

Limitations

PIE cannot calculate lift increments for configurations with five or more jets or configurations with three or more jets in a straight line either laterally or longitudinally. A temporary solution to this problem is to combined closely spaced jets into single jets, which can be rectangular or oval, until the total number of jets is reduced to four or less.

The methods used in this code are intended for use in preliminary design analysis and to give an indication of the effects of changes to primary configuration variables. Power induced effects are a complex function of many configuration variables and development of a V/STOL aircraft will require careful experimental investigations to accurately determine the induced forces (Reference 13).

Refer to Appendix B, the User's and Programmer's manuals, for additional information on configuration variables, how to run PIE, and more detail on the structure and organization of the code.

CHAPTER 5

Results and Discussion/Validation

When a new computer code is written there are invariably going to be flaws in the code due to calculation or logic errors. This is the reason all codes must to be verified and corrected before they are used reliably for the purposes they were created.

The purpose of this section is to compare the Power Induced Effects Module (PIE) with experimental results and results from Kuhn's programs. PIE is verified using height and forward velocity data from Kuhn's programs and experimental results. The other two independent variables, jet deflection angle and angle of attack, are not used to verify PIE due to lack of both experimental and computed data. It is difficult to isolate individual experimental lift increment from the total experimental lift increment which means only the total lift increments from PIE can be compared to experimental results. This is not a problem when verifying PIE results with Kuhn's programs. Since PIE and Kuhn's routines are based on the same methods, they can be compared lift increment for lift increment.

Many other conceptual configurations were tested that are not included in this report. These configuration were mainly used for error location purposes. They exercised the individual sections of the code to find blatant errors which caused PIE to abort or the lift increments to become unrealistic values. Only general trends and relative magnitudes in the lift increments were examined for these tests.

Configurations

The configurations used for verification start as simple flat plate models and progress in complexity to scale models of actual aircraft. These configurations are shown in Figure #16. The first four configurations, disk, body, wing-body and delta, are flat plate models which were experimentally tested in the Hover Test Rig at Rye Canyon by the STOVL/Powered Lift Division of NASA Ames (Reference 15). The next configuration, McDonnell Aircraft Company's (MCAIR) mixed-flow vectored thrust (MFVT), is a V/STOL design which is still in the preliminary design stages. The last configuration is a 15% scale model of an actual V/STOL aircraft, the YAV-8B (Harrier).

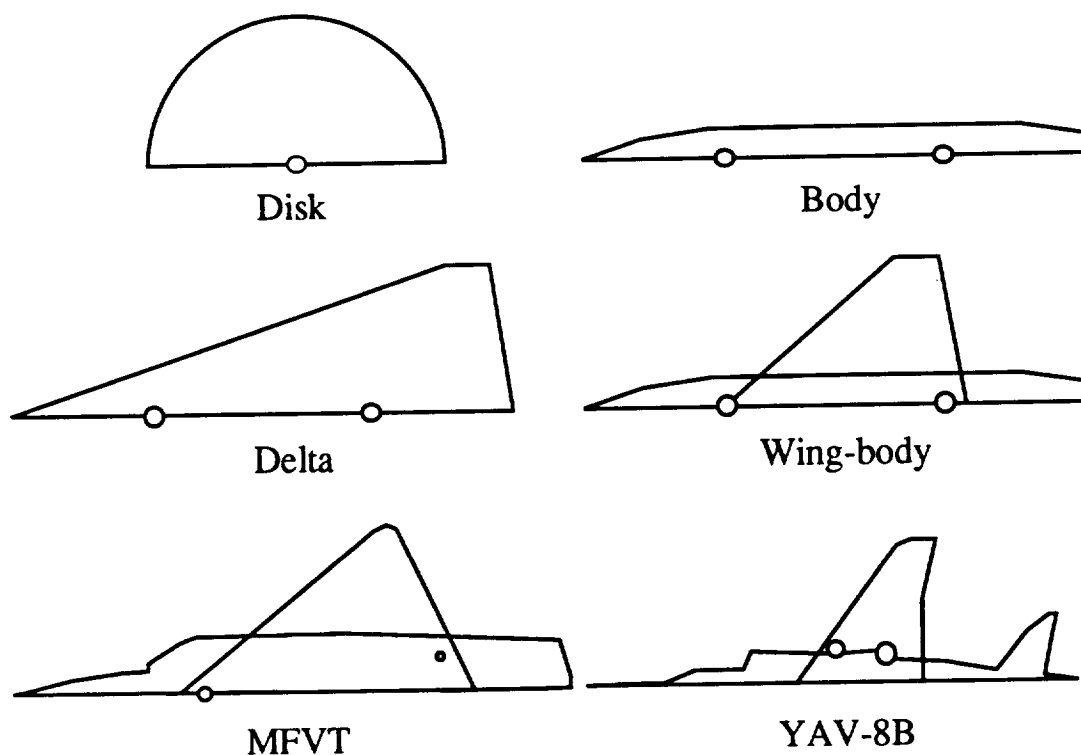


Figure #16. Half-planforms of configurations used for verification.

Flat Plate Models

Flat plate models have been experimentally hover tested for suckdown, fountain, and total lift increments encountered in and out of ground effect. The flat plate models tested were: disk, body, wing-body, and delta planforms.

The 20-inch circular disk was tested with one jet in the center. This configuration allowed the suckdown equations to be verified without the effect of any of the other lift increments. The body, wing-body, and delta planforms were tested with two jets placed longitudinally along the planforms. A fountain is created between the two jets on each of the last three planforms. This allows the fountain equations to be verified along with the suckdown equations.

Mixed-Flow Vectored-Thrust Configuration

The mixed-flow vectored-thrust (MFVT) configuration was proposed by the McDonnell Douglas, MCAIR division. Many different nozzle configurations were examined with the previous lift increment programs and considerable data is available. The MFVT was a conceptual design of a V/STOL aircraft that was examined with Kuhn's programs. Even though no experimental tests were conducted with the configuration, Kuhn's program results are valid because the PIE routines results are based on the same methods Kuhn's programs use.

The particular configurations examined in this report are a lateral two-jet case and a triangular three-jet case (one nozzle in front and two nozzles in the rear). The results from Kuhn's programs are very helpful due to the lack of experimental data for forward velocities.

YAV-8B

The YAV-8B, or harrier, configuration is the only configuration which has actual wind tunnel test data from a 15% scale model tested at the McDonnell Aircraft Company (Reference 5). In addition to hover data, a few forward velocity data points are available.

The YAV-8B also has some important configuration changes which were not fully tested in the previous cases. These are lift improvement devices (LIDS) on the underside of the aircraft which trap the fountain, and the jet flap which increases the overall positive circulation about the wing due to the position of the nozzles near the trailing edge of the wing. Both of these configuration changes add to the total lift increment. The experimental data (Reference 9) obtained is for the total lift increments, so only the total values are compared.

Hover

The hover calculations were conducted with no forward velocity, 90° jet deflection angle, and zero angle of attack. Most of the results were calculated from heights of two nozzle diameters to twenty nozzle diameters. The two nozzle diameter height was chosen because the method loses accuracy at very low heights due to a strong uncertain flow field created between the jets and the surface. The two nozzle diameter height is also chosen to be lower than the gear height for most aircraft configurations. The gear height is the height of the aircraft while it is on the ground with its gear extended. The twenty nozzle diameter maximum height was chosen because all lift increments either converge to zero or to a constant value. In Figures #17 to #21, the horizontal axis is labeled ' h/de ', where ' h ' is the actual height of the aircraft and ' de ' is the effective nozzle diameter or the diameter of a single nozzle which would have the same exit area as the sum of the areas of all nozzles on a configuration. This label refers to the number of effective nozzle diameters the aircraft is above the ground.

A general reason for the differences between PIE and experimental results for hover can be attributed to the assumption that all the lift increments are linear and can be summed together with no adjustment factors.

Disk

The 20-inch circular disk with a single jet is the simplest configuration tested. The curves in Figure #17 show the computed lift increment compared to the experimental lift increment. At altitudes larger than eight nozzle diameters the curves agree quite well, but for heights less than eight nozzle diameters the curves diverge slightly. This is most likely due to the large rate-of-change of the suckdown lift increment at low heights. Because of this large rate-of-change, any differences between the curves will cause sizable deviations between lift increment values at a given height.

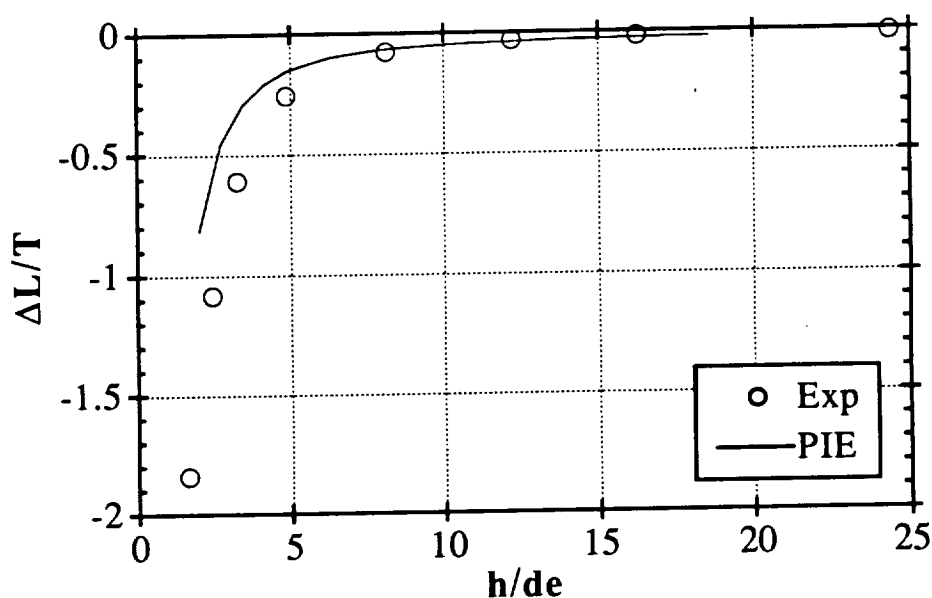


Figure #17. Comparison of calculated and measured total lift increments in hover for a 20-inch circular disk (Reference 15)

Body/Wing-Body

The body and wing-body configuration have two jets as opposed to the single jet present in the disk planform. Also the planform areas of the body and wing-body are much smaller than the planform area of the disk. These two facts cause the fundamental

differences between the magnitudes of the lift increments for the body, wing-body, and disk.

In Figure #18, PIE underpredicts the body total lift increment until a height of seven nozzle diameters is reached and overpredicts the total lift increment after seven nozzle diameters. The same underprediction of the lift increments at lower heights are found on the body as are found on the disk. This implies at lower heights the large slope of the curve magnifies any deviations between lift increment values. At greater heights, differences between experimental and predicted lift increments are not great and can be explained by inaccuracies in the methods for predicting the multiple jet suckdown at high altitudes. This is determined by examining the figures for the body, wing-body and delta planforms, all of which have multiple jets, where the lift increments at high altitudes are always overpredicted from the experimental results. Overall, the exact reasons for the differences are not entirely known. As mentioned by Kuhn (Reference 6), the multiple jet suckdown is the most important and most difficult element of the method to determine.

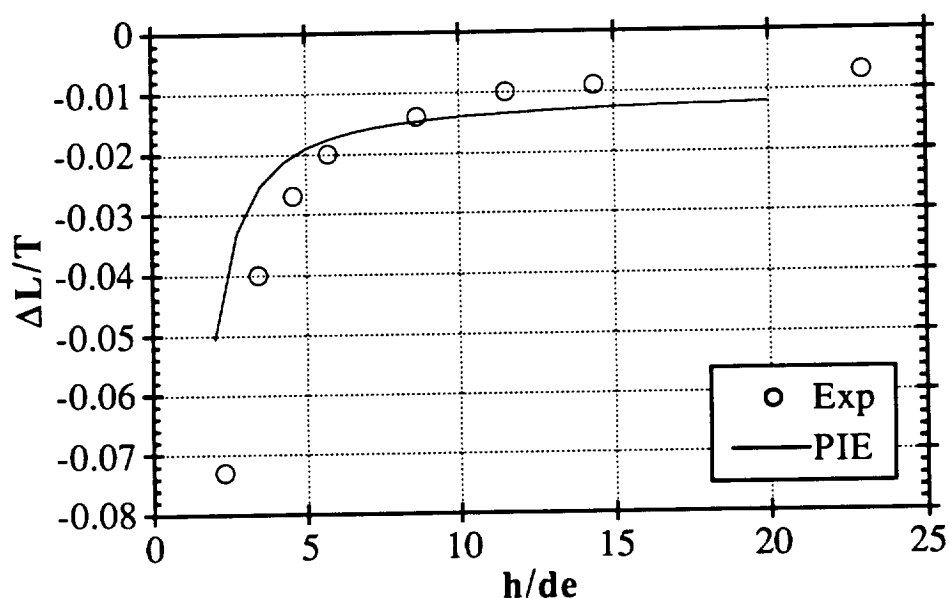


Figure #18. Comparison of calculated and measured lift increments in hover for the Body planform (Reference 15)

The calculated curve for the wing-body planform in Figure #19 matches closely with the experimental curve for altitudes less than 4.5 nozzle diameters. After this height, PIE overpredicts the lift increment. The good agreement for the lower altitudes is, most likely, a coincidence. Another problem with the method which could have caused the overprediction of the suckdown lift increment at higher altitudes could be the method was developed for low pressure ratios (less than 2.0) but was modified to extrapolate for higher pressure ratios. A higher jet pressure ratio (6.0) was used in the experimental tests. The extrapolation could have caused some differences. In general, higher pressure ratios will tend to entrain more flow around the configuration and thereby causing a greater suckdown increment. The higher pressure ratios also create stronger fountain increments due to the increase in strength of the lifting jets and therefore an increase in strength of the fountain.

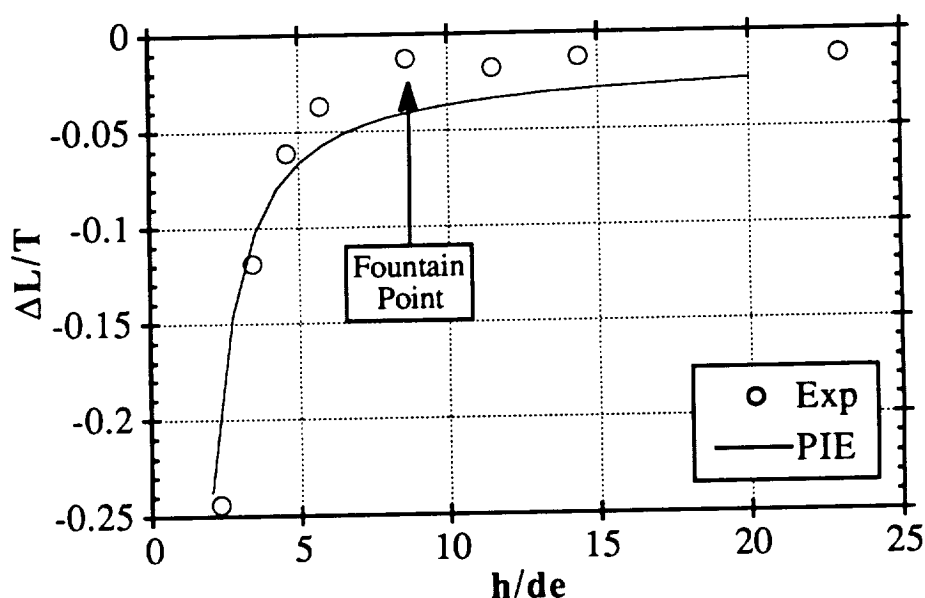


Figure #19. Comparison of calculated and measured lift increments in hover for the Wing-Body planform (Reference 15)

The point, labeled “fountain point”, is most likely an effect of the fountain created between the two jets. For configuration with larger fountains, a very perceptible hump is

usually created near this position in the curve. (A large hump can be seen in Figure #22 from the YAV-8B configuration.) It is also possible this point could be a stray data point from the test, but the first explanation is most probable. PIE does not predict this hump most likely because 1) the hump is in its beginning stages and 2) since PIE is overpredicting the suckdown, the hump is still being washed out before it starts.

Notice the magnitude of the lift increments for the body and wing-body (Figure #18 and #19) are much smaller than the lift increments for the circular disk (Figure #17). This is due the differences in area near the jet exits. Suckdown is caused by the decrease in pressure on the underside of a configuration from the entrainment action of a jet. Since pressure is force per area, then there would be a larger force on a configuration which has more area near the jet. The disk is inches in diameter which is five times larger than the body width of four inches and therefore the disk experiences a larger negative lift increment. This can also be seen by comparing the lift increments of the body and wing-body configurations. The wing-body has a larger lift increment magnitude because it has the wing and the body near the nozzles. Basically the wing area near the nozzles causes the increase in the suckdown lift increment over that of the body.

Delta

The calculations for the delta planform agree well with the experimental data from the hover test rig. The two curves in Figure #20. are almost identical for low altitudes and diverge by only a small amount at the higher altitudes. Also, as expected, the magnitude of the lift increments were higher than the wing-body increments, due to the increase in area around the nozzles of the delta planform.

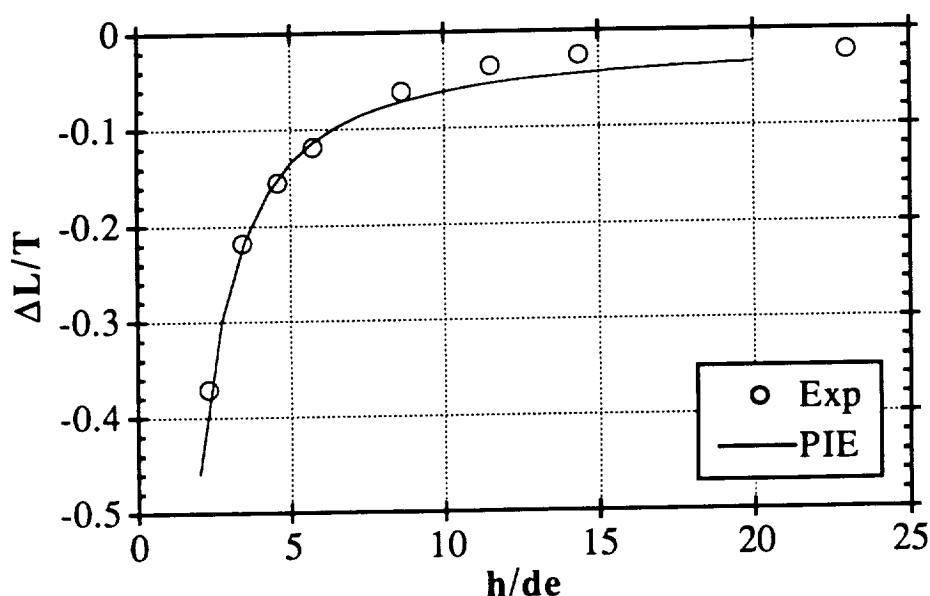


Figure #20. Comparison of calculated and measured lift increments in hover for the Delta planform (Reference 15)

Mixed Flow Vectored Thrust

The MFVT configuration with 3-jets is examined because a two jet cases was examined with the flat plate tests and the YAV-8B tests examine the 4-jet case. Figure #21 is a comparison between the original program developed by Kuhn and PIE. As can be seen, all the lift increments are approximately the same except for a slight divergence between the suckdown terms. This deviations between the two codes can be attributed to the differences in the configuration variables. A great effort was made to ensure all configurations variables were the same throughout each program, but there were some variables which were calculated differently in PIE than were input by the user of Kuhn's program. These variables included the planform areas and \bar{D} , the angular mean diameters of the planforms.

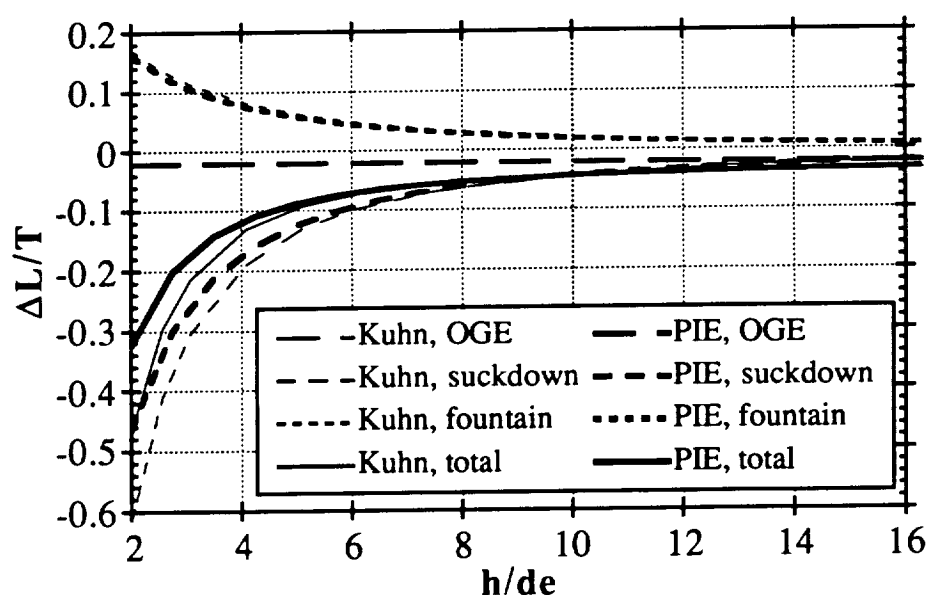


Figure #21. Comparison of calculated lift increments in hover between Kuhn's program and PIE for the MFVT, triangular 3-jet configuration (Reference 7)

YAV-8B

Figure #22 is a comparison between the total lift increments from wind tunnel tests of a 15% scale model of the YAV-8B and the total lift increments calculated by PIE. The hump in the data at the beginning of the graph is a result of the fountain arms and core caused by the four lifting jets of the configuration. The fountain effect tapers off at heights around 15 ft. for the calculated curve. The experimental curves has some fountain effects up to 24 ft. After the fountain effect diminishes, both curves reflect the trend of the suckdown increments. Some of the differences at the higher heights could be explained by the methods which were developed for the effect of lift improvement devices, wing height and body contour. (The body contour of some configurations are typically rounded to some extent, especially on the YAV-8B. Since the experimental results were found mostly

from test data of flat plate models, a multiplication factor was developed to correct for the flat plate data.) The development of methods for these effects were all based on very

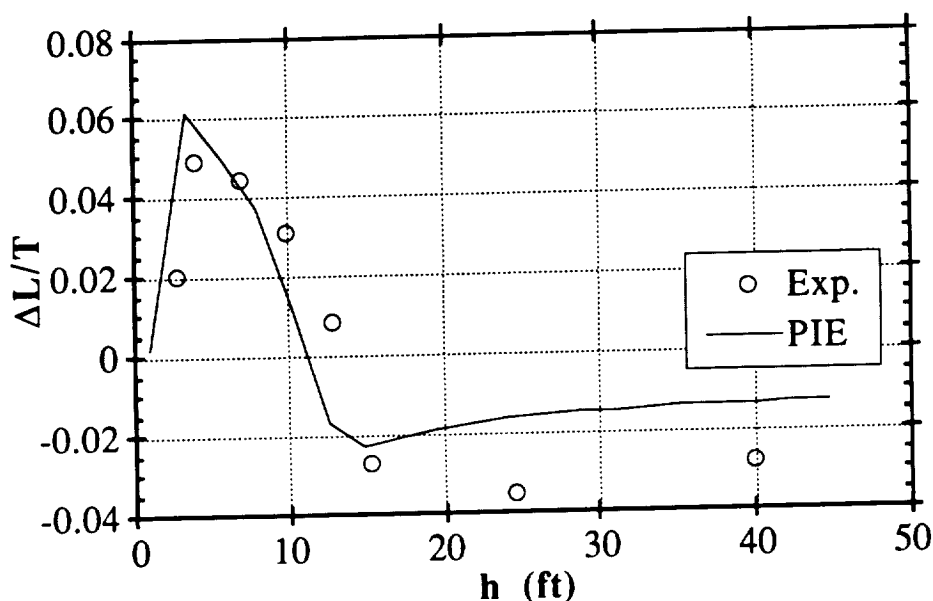


Figure #22. Comparison of calculated and measured total lift increments in hover for 15% scale model of the YAV-8B (Reference 9)

limited experimental data and therefore more experimental tests need to be completed to refine and gain confidence in the methods. Overall the results are promising for this case.

Forward Flight

Two additional lift increments are calculated for forward flight with nozzle deflection than are calculated for hover. These are the ground vortex and jet wake increments. Since forward flight contains all lift increments, it should be carefully verified. The only problem is experimental data for forward flight cases are very limited. The only actual experimental data available is for the YAV-8B. Due to the lack of data, more testing should be done to further validate the forward flight predictions.

Forward flight with nozzle deflection is an important flight regime for V/STOL aircraft. This typically occurs when a V/STOL aircraft is in transition from jet bourn to

wing bourn flight or visa versa. Transition usually occurs near the ground and is a critical time for the aircraft. It can be a great advantage to the aircraft designer, while working on preliminary design, to know the lift increment characteristics of the aircraft in transition. This allows the designer to modify their configuration to obtain the most favorable forward flight lift increment characteristics.

MFVT

Again, the programs developed by Kuhn were used to check the validity of PIE for the MFVT 2-jet and 3-jet cases. The graphs created for this section were created to observe the variation of the lift increment with respect to V_e , the dynamic pressure ratio. The dynamic pressure ratio is the square root of the ratio of the dynamic pressure of the aircraft to the dynamic pressure of the jet.

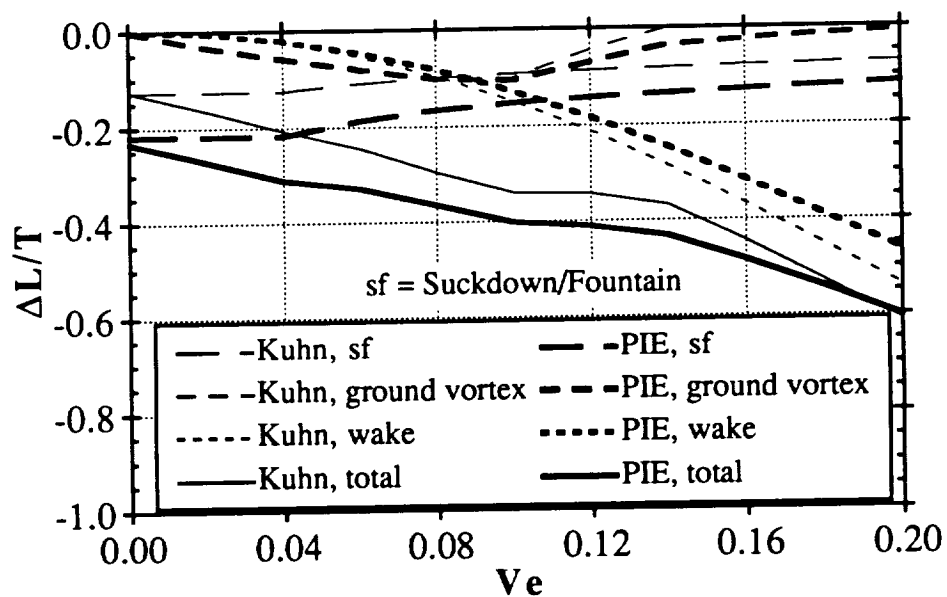


Figure #23. Comparison of calculated lift increments in forward flight at an altitude of 6 ft. for the MFVT, lateral 2-jet configuration from Kuhn's program and PIE (Reference 7)

Figure #23 is a comparison of the lift increments calculated from PIE and Kuhn's program for the MFVT configuration, with two laterally placed jets, in forward flight. There are a number of discrepancies in the two codes which are explained next.

PIE overpredicts the suckdown/fountain from Kuhn's program by almost 80% at low velocities. The magnitude of overprediction decreases as the velocity is increased. The reason for this overprediction is when Kuhn's program calculates this term, it uses a single jet equation, equation (4) without the multiple jet factor K_S , to calculate the multiple jet suckdown. This single jet equation calculates a much smaller suckdown effect than the multiple jet equation. Another small discrepancy in the suckdown calculation is the addition of an extra OGE suckdown term, equation (3), present in the fountain calculation. This would tend to offset the previous miscalculation, but the magnitude of the OGE term is significantly less than the suckdown term in ground effect.

The ground vortex terms agree closely until the dynamic pressure ratio (V_e) reaches a value of .1. After this point Kuhn's calculations tend towards zero faster than PIE's calculations. This is attributed to the variation in the algorithms between the two codes. The maximum height for which the ground vortex effects are felt is not the same for the body as for the wing. Kuhn's code assumes the total ground vortex term is zero when the maximum ground vortex height for the body is reached. The method described in reference 12 indicates the ground vortex effects still occur on the wing after the effects on the body reach zero. From the graph it can be seen when Kuhn's ground vortex term reaches zero, PIE's ground vortex term changes slope. This is because the ground vortex effects are still occurring on the wing after the body's ground vortex effects become zero. This is the method is presented in reference 13.

The wake terms also are the same at low velocities and diverge with higher velocity. These differences can be attributed to three reasons:

- 1) Kuhn's program does not include the lift adjustment factors for longitudinal position and flap extension which are used with equation (16).

- 2) A mistake in the formula for the wing aspect induced lift adjustment factor was found in Kuhn's code.
- 3) A term is used in Kuhn's code which is only to be calculated when there are longitudinal placed jets. This configuration has laterally placed jets.

Figure #24 is a comparison of lift increments calculated from PIE and Kuhn's program for the MFVT configuration, with three jets, in forward flight. The discrepancies found in this figure for the suckdown/fountain and the ground vortex lift increments are due to the same reasons suggested above for the two-jet case.

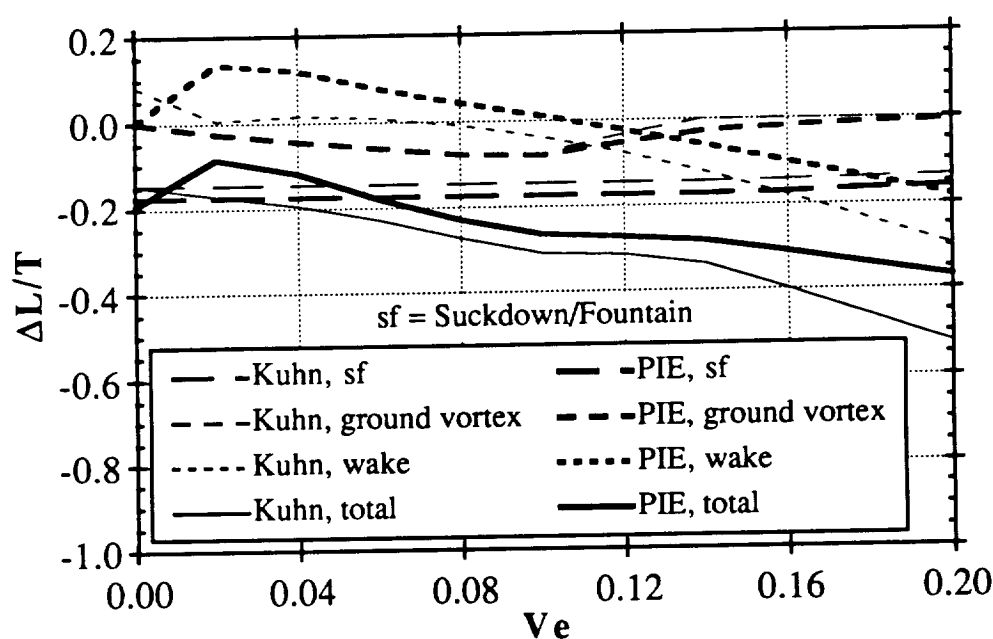


Figure #24 Comparison of calculated lift increments in forward flight at an altitude of 6 ft. for the MFVT, triangular 3-jet configuration from Kuhn's program and PIE (Reference 7)

The differences between the wake terms are also for the same reason as the two-jet case except there are a few other distinctions. When Kuhn's program calculates the aspect induced lift adjustment factor, which is used with equation (16), it uses the lowest body width instead of the width at the wing which is the width PIE uses. PIE does not use the lowest body width so therefore the values are perturbed slightly. Another difference

between the two codes, is when PIE calculates the longitudinal position adjustment factor for the wing, it uses an equation specific for the wing. Kuhn's program uses a value of 1.0 for the wing. Also when there is a jet near the trailing edge, Kuhn's program uses a different equation for the lift increment than the wing equation, similar to equation (17), used by PIE. This is the reason for the different trends during the low velocities of the graph. This wing equation calculates a positive value for the wake lift increment due to the placement of the jets in relation to the trailing edge of the wing (the jet flap effect).

The total lift increments for each configuration follow from the summation of all the individual lift increments. Since each individual lift increment from the two codes are not the same then it follows the total lift increments will not be identical either.

The comparison between the two codes helped debug PIE and also helped find some discrepancies between the original programs developed by Kuhn and the methods developed by Kuhn and Stewart.

YAV-8B

The forward velocity experimental data for the YAV-8B is very limited, and provides only three velocity points; 0, 17 and 24 knots. These correspond to dynamic pressure ratios of 0.0, 0.023 and 0.033 respectively. More experimental data points are needed, especially in the higher velocity region, to draw more meaningful conclusions.

The comparisons between experimental and calculated lift increments for the YAV-8B are shown in Figure #25. Three different altitudes are shown in the graph: 4, 10, and 25 ft. The trends of the experimental test can not be accurately determined from three data point so the only analysis which can be accomplished is to examine the magnitudes of the lift increments. The lift increments at lower heights tend to agree more than those at greater heights but for the limited velocity range, PIE predicted the total lift increments well. A possible reason for the differences between PIE and experimental results can be attributed to the assumption that all lift increments are linear and can be summed together

with no adjustment factors. (The experimental data in Figure #25 are connected with lines to allow the reader to distinguish between the three groups of curves).

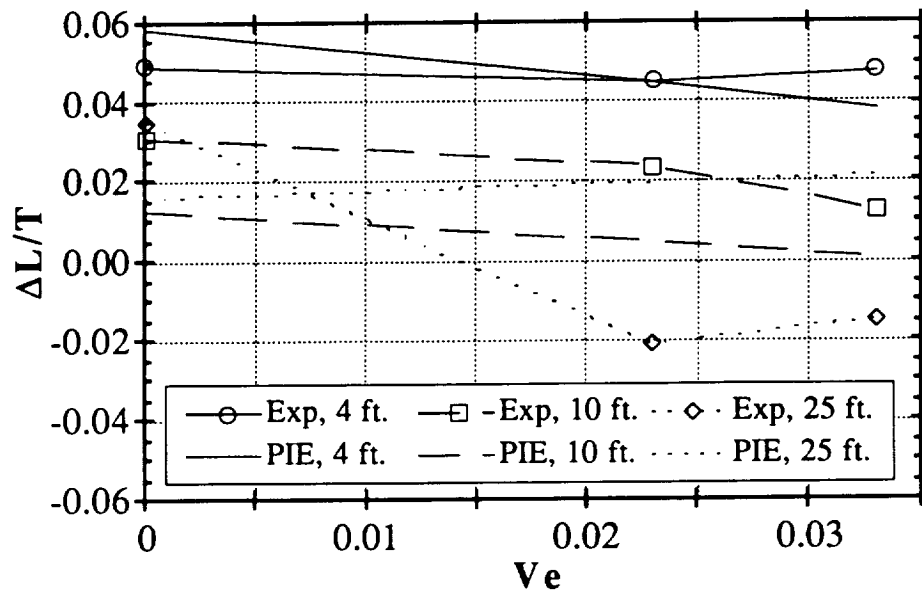


Figure #25. Comparison of calculated and measured total lift increments in forward flight for a 15% scale model of the YAV-8B (Reference 9)

CHAPTER 6

Conclusions and Recommendations

An easy to use, and modify computer code was developed which makes preliminary estimates of lift increments encountered by V/STOL aircraft due to the suckdown, fountain, ground vortex, and jet wake. This code provides the user with a quick and easy-to-use way to evaluate the trends and magnitudes of lift increments encountered by V/STOL aircraft in hover and transition.

The Power Induced Effects Module is able to reproduce the trends of the lift increments encountered in hover accurately for the flat plate models. The magnitude of the lift increments are reasonably accurate for the range of values investigated. The values of the MFVT hover lift increments calculated using Kuhn's program and PIE agree quite well and any inaccuracies are explained by the differences between the configuration variables. The hover lift increments calculated by PIE for the YAV-8B agree well with the experimental results.

The lift increments encountered in forward flight are calculated by PIE and Kuhn's code for two MFVT configurations. The magnitudes and trends of these lift increments agree well except for small differences which are caused by slight deviations between the algorithms of the two programs. There are not enough experimental lift increment points to accurately evaluate the trends from PIE for the YAV-8B. The magnitudes of the calculated lift increments fall close to those of the experimental lift increments. Given the number of configuration variables, these accuracies are quite good.

The most obvious drawback of the power induced effect module is the lack of forward flight, jet deflection angle, and angle of attack validation. Further validation of

PIE needs to be accomplished for these independent variables. Also the code needs to be fully incorporated into the ACSYNT design program so it can be used to its fullest potential.

An enhancement to the code would be the addition of routines which would calculate the moment increments which are caused by the non-symmetric lift increments. This would not be very difficult because the code was initially written with the intent to include these moment increments so most of the necessary variables have already been incorporated within the code. Lack of time was the reason this was not done for this project. Much testing and validation needed to be completed for the moment increment routines, similar to the amount done for the lift increment routines.

BIBLIOGRAPHY

1. Aircraft Synthesis Program (ACSYNT), National Aeronautics and Space Administration, Ames Research Center, Moffett Field, California, 1975.
2. Cimbala, J.M., Billet, M.L. and Gaublumme, D.P. "Experiments on the Unsteady Ground Vortex." NASA Contractor Report 177566. p. 1, August, 1990.
3. Hammond, A.D. "Thrust Losses in Hovering for Jet VTOL Aircraft." Paper presented at the Conference on V/STOL and STOL Aircraft, Ames Research Center, Moffett Field, California. NASA SP-116, p. 163. April 4-5, 1966.
4. Henderson, C., Clark, J., Walters, M. "V/STOL Aerodynamics and Stability & Control Manual" Naval Air Development Center, Pennsylvania, January 15, 1980. NADC-80017-60.
5. Holl, D.M., Hollingsworth, E.G. and Lacey, T.R. "AV-8B V/STOL Wind Tunnel Tests in Light of Flight Experience." AIAA Aerospace Engineering Conference & Show, Los Angeles, California. AIAA-90-1817. February 13-15, 1990.
6. Kuhn, R.E. "An Engineering Method of Estimating the Induced Lift on V/STOL Aircraft Hovering In and Out of Ground Effect." V/STOL Consultant. NADC-80246-60. pp. January, 1981.
7. Kuhn, R.E. "HOVER-GE", "JET-IND2", "RCS-IND", "STOL-GE2", "STOL-GV2", "STOL-SF3", "STOL-WK3" (BASIC computer programs) V/STOL Consultant. 1989-1990.
8. Kuhn, R.E. "V/STOL and STOL Ground Effects and Testing Techniques." V/STOL Consultant. NASA Report NAS-11912, December, 1984.
9. "Longitudinal Jetborne Characteristics & Cross-Wind Data for Harrier II, From 'AV-8B Aircraft Stability, Control, and Flying Qualities.'" MDC A5192. Revision A. October 15, 1980, pp. 4.150 - 4.167.
10. Margason, R.J. "Jet-Induced Effects in Transition Flight." Paper presented at the Conference on V/STOL and STOL Aircraft, Ames Research Center, Moffett Field, California. NASA SP-116. p 179. April 4-5, 1966.
11. Margason, R.J. "Review of Propulsion-Induced Effects on Aerodynamics of Jet/STOL Aircraft." NASA TN D-5617. February, 1970.
12. Raymer, D.P. Aircraft Design: A Conceptual Approach, AIAA Educational Series, Washington D.C., AIAA, Inc., pp. 540-541, 1989.

13. Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the Aerodynamic Stability and Control Parameters of STOL Aircraft Configurations." North American Aircraft Operations. Rockwell International Corporation. AFWAL-TR-87-3019. Volume II & III. June 1987.
14. Vogler, R.D. "Surface Pressure Distributions Induce on a Flat Plate by a Cold Air Jet Issuing Perpendicularly from the Plate and Normal to a Low-Speed Free-Stream Flow." NASA TN D-1629. pp. 1, 20. March, 1963.
15. Wardwell, D.A. "Hover Test Rig at Rye Canyon, CA (Computerized data)." STOVL/Powered Lift Division (FAP), NASA Ames Research Center. 1989

APPENDIX A

PIE FORTRAN Code

Coppie

PROGRAM COPPIE

```

C-----
C ACRONYM:  Control Program for Powered Induced Effects.
C          --      -      -      -      -
C PURPOSE:  The purpose of COPPIE is to act as the control program (or
C           driver) for the Power Induced Effects Module.  Its function
C           is to know everything about the aircraft that is being
C           analyzed.  In other words, it will provide all the
C           subroutines that it calls with what what ever input is
C           needed to complete the calculations for a particular
C           configuration.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   ICALC      I    -    ----      Flag which ACSYNT uses to control
C                                     execution of subroutines.
C                                     1 = Input
C                                     2 = Calculations
C                                     3 = Output
C                                     (Not used at this time)
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C (This variable list contains all variables in the Power Induced
C Effects module separated by their respective common blocks.  The rest
C of the subroutines contains only those variables which are present
C within that individual subroutine.)
C-----
C CONPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   AEND       R    I    deg      Ending value for angle of attack
C   ASTART     R    I    deg      Starting value for angle of attack
C   ASTEP      R    I    deg      Step value for angle of attack
C   DEND       R    I    deg      Ending value for jet deflection
C                                     angle
C   DSTART     R    I    deg      Starting value for jet deflection
C                                     angle
C   DSTEP      R    I    deg      Step value for jet deflection angle
C   HEND       R    I    Ft       Ending value for altitude
C   HSTART     R    I    Ft       Starting value for altitude
C   HSTEP      R    I    Ft       Step value for altitude
C   VEND       R    I    kts      Ending value for aircraft velocity
C   VSTART     R    I    kts      Starting value for aircraft

```

C					velocity
C	VSTEP	R	I	kts	Step value for aircraft velocity
C	-----				
C	FIGPIE Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----				
C	B_B	R	I	Ft	Width of Body
C	B_CS	R	I	Ft	Width of Center Section
C	B_JP	R	I	Ft	Width of Jet Pattern
C	B_W	R	I	Ft	Width of Wing (Wing span)
C	B_WB	R	I	Ft	Width of Wing-Body
C	CONFIG	C*18	I	----	Short title of current config
C	D_F	R	I	Ft	Diameter of each Front jet
C	D_R	R	I	Ft	Diameter of each Rear jet
C	D_RCS	R	I	Ft	Diameter of Roll RCS nozzle
C	DB_B	R	I	Ft	Dbar of Body
C	DB_CS	R	I	Ft	Dbar of Center Section
C	DB_W	R	I	Ft	Dbar of Wing
C	DB_WB	R	I	Ft	Dbar of Wing-Body
C	DE_F	R	I	Ft	Effective Diameter of Front jets
C	DE_FR	R	I	Ft	Effective Diameter of Front & Rear jets combined
C	DE_R	R	I	Ft	Effective Diameter of Rear jets
C	DR	R	I	----	Density Ratio (Jet / Atm)
C	E_1	R	I	Ft	Half dist between adjacent jets (1)
C	E_3	R	I	Ft	Half dist between adjacent jets (3)
C	E_4	R	I	Ft	Half dist between adjacent jets (4)
C	F_NAME	C*50	I	----	Name of file which contains body & wing planform coordinates
C	KB	R	I	----	Boundary layer factor
C	KLF	R	I	----	Adjustment factor for flap extension when calculating the lift loss due to jet induced effects.
C	KR	R	I	----	Body contour factor
C	L_B	R	I	Ft	Length of body
C	L_CS	R	I	Ft	Length of Center Section
C	L_F	R	I	Ft	Length of Front jet
C	L_R	R	I	Ft	Length of Rear jet
C	L_WB	R	I	Ft	Length of Wing-Body
C	MAC	R	I	Ft	Mean Aerodynamic Chord of wing
C	MD_RCS	R	I	lbm/s	Mass flow rate for one RCS nozzle
C	N_BODY	I	I	----	Number of data points for Body planform
C	N_CS	I	O	----	Number of data points for Center Section planform
C	N_WING	I	I	----	Number of data points for Wing planform
C	N_WB	I	I	----	Number of data points for Wing-Body planform
C	NDIV	I	I	----	Number of divisions to be used when calculating suckdown areas (SDAREA) and Dbars (DIABAR)
C	NUM	I	I	Ft	Total NUMBER of jets
C	NUM_F	I	I	----	NUMBER of Front jets
C	NUM_R	I	I	----	NUMBER of Rear jets
C	PER_FR	R	I	Ft	Total perimeter of jets
C	PP_LID	R	I	----	Ratio of perimeter enclosed by lids to total perimeterC
C					

C	PR_F	R	I	----	Jet Pressure Ratio for front jets
C	PR_FR	R	I	----	Jet Pressure Ratio for all jets
C	PR_R	R	I	----	Jet Pressure Ratio for rear jets
C	PR_RCS	R	I	----	Roll RCS Pressure Ratio
C	PT_RCS	R	I	lb/sq ft	Total pressure for one roll RCS jet
C	Q_F	R	I	lb/sq ft	Dynamic pressure for the front jets
C	Q_FR	R	I	lb/sq ft	Dynamic pressure for Front and Rear jets
C					
C	Q_R	R	I	lb/sq ft	Dynamic pressure for the rear jets
C	Q_RCS	R	I	lb/sq ft	Dynamic pressure for roll RCS jets
C	R_B	R	I	Ft	Corner radius of body sides.
C	S_B	R	I	Sq Ft	Area of Body
C	S_CS	R	I	Sq Ft	Area of Center Section
C	S_F	R	I	Sq Ft	Area of Front jets
C	S_FA1	R	I	Sq Ft	Area affected by fountain arm (1)
C	S_FA3	R	I	Sq Ft	Area affected by fountain arm (3)
C	S_FA4	R	I	Sq Ft	Area affected by fountain arm (4)
C	S_FR	R	I	Sq Ft	Total jet exit area
C	S_JP	R	I	Sq Ft	Area enclosed by Jet Pattern
C	S_LID	R	I	Sq Ft	Area enclosed by LIDS
C	S_R	R	I	Sq Ft	Area of Rear jets
C	S_W	R	I	Sq Ft	Area of Wing
C	S_WB	R	I	Sq Ft	Area of Wing-Body
C	SCALE	R	I	----	Actual scale factor which adjust the area of fountain arm (3) to account for the fact the curvature of the aircraft's nose tends not to hold fountain arm very well causing the area that fountain arm affects to be scaled down.
C					
C					
C					
C					
C					
C	SCALE3	R	I	----	Fountain arm (3) scaler (Percentage measured from the front jets to the nose of the aircraft.) Tries to approximate SCALE but does not do a great job (better than nothing.)
C					
C					
C	SF_FB	R	I	Sq Ft	Area ahead of Front jets using body planform
C	SF_FW	R	I	Sq Ft	Area ahead of Front jets using wing planform
C	SF_FWB	R	I	Sq Ft	Area ahead of Front jets using the wingbody planform
C	SF_RB	R	I	Sq Ft	Area ahead of Rear jets using the body planform
C	SF_RW	R	I	Sq Ft	Area ahead of Rear jets using the wing planform
C	SF_RWB	R	I	Sq Ft	Area ahead of Rear jets using the wingbody planform
C					
C	SP_FA1	R	I	Sq Ft	Potential area affected by fountain arm (1)
C	SP_FA3	R	I	Sq Ft	Potential area affected by fountain arm (3)
C	SP_FA4	R	I	Sq Ft	Potential area affected by fountain arm (4)
C					
C	SP_JP	R	I	Sq Ft	Actual surface area within area enclosed by nozzles
C					
C	SPLY_F	R	I	Ft	SPLaY angle of Front jet
C	SPLY_R	R	I	Ft	SPLaY angle of Rear jet
C	T_F	R	I	lb	Thrust of Front jets

C	T_FR	R	I	lb	Total thrust in all jets
C	T_R	R	I	lb	Thrust of Rear jets
C	T_RCS	R	I	lb	Thrust of roll RCS nozzle
C	THA_1	R	I	deg	Half angle between jets (1)
C	THA_3	R	I	deg	Half angle between jets (3)
C	THA_4	R	I	deg	Half angle between jets (4)
C	TP_RCS	R	I	Rankine	Temperature of flow in roll RCS nozzle
C	TT_F	R	I	----	Front Thrust / total Thrust
C	TT_R	R	I	----	Rear Thrust / total Thrust
C	W_F	R	I	Ft	Width of Front jet
C	W_R	R	I	Ft	Width of Rear jets
C	WIWE	R	I	----	Flow Weight of Inlet / Flow Weight of Exit
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_CS	R	I	----	Width to Length ratio of Center Section
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_JP	R	I	----	Width to Length ratio of Jet
Pattern					
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	WL_WB	R	I	----	Width to Length ratio of Wing-Body
C	X_BODY(500)	R	I	Ft	X-coordinates of Body planform
C	X_CORE	R	I	FT	X-coordinate of fountain core
C	X_CS(30)	R	O	Ft	X-coordinates of Center Section planform
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	X_RCS	R	I	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
C	X_WB(500)	R	I	Ft	X-coordinates of Wing-Body planform
C	X_WING(500)	R	I	Ft	X-coordinates of Wing planform
C	XCA_CG	R	I	Ft	Distance of center of area ahead of CG
C	X_CA	R	I	Ft	X-coordinate of Center of Area
C	X_CG	R	I	Ft	X-coordinate of Center of Gravity
C	XCG_C2	R	I	Ft	Distance from CG to MAC/2
C	XCG_C4	R	I	Ft	Distance from CG to MAC/4
C	XCG_F	R	I	Ft	Distance Front jet is ahead of CG
C	XCG_I	R	I	Ft	Inlet longitudinal distance ahead of CG
C	XCG_R	R	I	Ft	Distance Rear jet is ahead of CG
C	XNOZ_F	R	I	Ft	X-coordinates of Front NOzzle
C	XNOZ_R	R	I	Ft	X-coordinates of Rear NOzzle
C	XTE_F	R	I	Ft	X-coordinate of trailing edge for front nozzle (root TE if Nozzle within body)
C	XTE_R	R	I	Ft	X-coordinate of trailing edge for rear nozzle (root TE if Nozzle within body)
C	Y_1	R	I	Ft	Spanwise extent of planform on fountain arm (1) center line.
C	Y_3	R	I	Ft	Spanwise extent of planform on fountain arm (3) center line.
C	Y_4	R	I	Ft	Spanwise extent of planform on fountain arm (4) center line.
C	Y_BODY(500)	R	I	Ft	Y-coordinates of Body planform
C	Y_CORE	R	I	FT	Y-coordinate of fountain core
C	Y_CS(30)	R	O	Ft	Y-coordinates of Center Section

C					planform
C	Y_F	R	I	Ft	Distance between Front jets
C	YNOZ_F	R	I	Ft	Y-coordinates of Front NOZZle
C	YNOZ_R	R	I	Ft	Y-coordinates of R NOZZle
C	Y_R	R	I	Ft	Distance between Rear jets
C	Y_RCS	R	I	Ft	Distance of roll RCS nozzle in from wingtip
C					Y-coordinates of Wing-Body planform
C	Y_WB(500)	R	I	Ft	Y-coordinates of Wing planform
C	Y_WING(500)	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	YB_F	R	I	Ft	Lateral distance from Body to Rear jets (for external jets)
C	YB_R	R	I	Ft	Max spanwise extent of planform (1)
C	YP_1	R	I	Ft	Max spanwise extent of planform (3)
C	YP_3	R	I	Ft	Max spanwise extent of planform (4)
C	YP_4	R	I	Ft	Lateral distance from Wingbody to Front jets (for external jets)
C	YWB_F	R	I	Ft	Lateral distance from wingbody to Rear jets (for external jets)
C	YWB_R	R	I	Ft	Height of body base above nozzle
C	Z_B	R	I	Ft	Inlet vertical distance above CG
C	ZCG_I	R	I	Ft	height of wing above nozzle
C	Z_W	R	I	Ft	
C	HOVPIE Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C					
C	AJ	R	I	Sq Ft	Total jet exit area.
C	BF1	R	I	Ft	Half length of fountain arm (1).
C	BF3	R	I	Ft	Half length of fountain arm (3).
C	BF4	R	I	Ft	Half length of fountain arm (4).
C	BFP1	R	I	Ft	Maximum spanwise extent of fountain arm (1).
C	BFP3	R	I	Ft	Maximum spanwise extent of fountain arm (3).
C	BFP4	R	I	Ft	Maximum spanwise extent of fountain arm (4).
C	DB	R	I	Ft	Body D bar.
C	DC	R	I	Ft	Configuration D bar.
C	DE	R	I	Ft	Equivalent diameter of jets.
C	DH	R	I	Ft	Delta height of wing.
C	E	R	I	Ft	Length to width ratio of jet pattern.
C	HEIGHT	R	I	Ft	Altitude at which calculations are to be made.
C	HW	R	I	Ft	Half width of body for jets outside outside of body, else use half distance between adjacent jets.
C	KRB	R	I	Ft	Body contour factor
C	L	R	I	Ft	Length of configuration.
C	NJETS	I	I	----	Total number of jets on config
C	PER	R	I	Ft	Total perimeter of jets (all jets)
C	PP	R	I	----	Fraction of lid perimeter enclosed
C	PR	R	I	----	Jet pressure ratio.
C	SB	R	I	Sq Ft	Planform area of body.
C	SC	R	I	Sq Ft	Area enclosed by jet pattern.
C	SCP	R	I	Sq Ft	Actual surface area inside jet pattern.
C					

C	SFA1	R	I	Sq Ft	Area affected by fountain arms (1).
C	SFA3	R	I	Sq Ft	Area affected by fountain arms (3).
C	SFA4	R	I	Sq Ft	Area affected by fountain arms (4).
C	SFA1P	R	I	Sq Ft	Potential surface area between jets (1)
C					Potential surface area between jets (3)
C	SFA3P	R	I	Sq Ft	Potential surface area between jets (4)
C	SFA4P	R	I	Sq Ft	Area enclosed by lids.
C	SL	R	I	Sq Ft	Planform area of configuration
C	SP	R	I	Sq Ft	Front jet splay angle.
C	SPLAY	R	I	Deg	Half angles between jets (1).
C	THA1	R	I	Deg	Half angles between jets (3).
C	THA3	R	I	Deg	Half angles between jets (4).
C	THA4	R	I	Deg	Width of body.
C	WB	R	I	Ft	Width of configuration.
C	WC	R	I	Ft	Half distance between adjacent jets (1).
C	X1	R	I	Ft	Half distance between adjacent jets (3).
C					Half distance between adjacent jets (4).
C	X3	R	I	Ft	Distance center of area ahead of CG
C	X4	R	I	Ft	Distance between front jets.
C	XCA	R	I	Ft	Distance between rear jets.
C	YF	R	I	Ft	
C	YR	R	I	Ft	
C	-----				
C	MISC Common Block				DESCRIPTION
C	NAME	TYPE	I/O	UNITS	-----
C	POINTS	I	I	----	Number of points which define polygon (not to exceed 500)
C	XPTS(500)	R	I	----	X-coordinates of polygon
C	YPTS(500)	R	I	----	Y-coordinates of polygon
C	TOTAL	R	I	----	Total area enclosed by the polygon.
C	-----				
C	PIEFLAG Common Block				DESCRIPTION
C	NAME	TYPE	I/O	UNITS	-----
C	FLGRCS	L	O	----	Flag which signals if there is an RCS on the configuration (TRUE - RCS, FALSE - No RCS)
C	HDEOUT	L	I	----	Signals when to output tables based on height or height/De TRUE - Print based on Height/DE, FALSE - Print based on height
C	PRTFLG	L	I	----	Signals when to output to screen.
C	RCSFLG	L	O	----	Flag which identifies if RCS lift loss has calculation exceeded the area ratio (Swing / Ajet > 7000) TRUE - Exceeded FALSE - Not Exceeded
C	TYPE_F	C*4	O	----	Description of type of front nozzle (CIRCular, OVAL, RECTangular)
C	TYPE_R	C*4	O	----	Same as TYPE_F but for rear nozzles
C	VEOUT	L	I	----	Signals when to output tables based on VE TRUE - Print Based on VE

FALSE - Print Based on Velocity
Identifies when WingBody planform
has been enter by the user. Used to
determine when to use wingbody in
calculations.

PIEGV Common Block					DESCRIPTION
NAME	TYPE	I/O	UNITS		
WFLAG	L	O	----		
AFACT	R	I	----		Area ratio - based on wether the wing or body is being used
AJ	R	I	Sq Ft		Area of the front jets
DE	R	I	Ft		Equivalent Diameter of front jets
DF	R	I	Ft		Diameter of front jets
DH	R	I	Ft		Height of wing above jets
EX	R	I	Ft		Distance from jet pattern center to center of front jets.
LTV	R	O	----		Lift loss due to ground vortex.
N	I	I	----		Number of front jets
Q	R	I	Lb/sq ft		Dynamic pressure of the nozzles
TFT	R	I	----		Front thrust split
WLB	R	I	----		Width to length ratio of the body.
WLJ	R	I	----		Width to length ratio of the jets.

PIERCS Common Block					DESCRIPTION
NAME	TYPE	I/O	UNITS		
B	R	I	feet		Wing span
DRCS	R	I	feet		Roll RCS jet Diameter.
NPR	R	I	----		Roll jet nozzle pressure ratio.
Q	R	I	lb/sq ft		Roll jet dynamic pressure.
S	R	I	sq ft		Wing area
SAJ	R	I	----		Wing area / jet area
T	R	I	lb		RCS roll jet thrust
X	R	I	feet		Jet distance ahead of wing trailing edge.
Y	R	I	feet		Jet distance in from wing tip.

PIERROR Common Block					DESCRIPTION
NAME	TYPE	I/O	UNITS		
DBERR	I	O	----		Error flag returned by DIABAR. If IFLAG = 1, then an explanation of the problem is given at the time of the error.

0 = no errors.
1 = values of dbar & area probably bad.
2 = moved x-position of dbar point/nozzle point to geometric center.
3 = input NANGLE was inappropriate (set to absolute value or 1000).
4 = Points entered in wrong direction, routine aborted. Must enter points from left to right.

C PIEWK Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	AFACT	R	I	----	Area ratios used in calculating the lift loss due to the jet wake.
C	AJ	R	I	Sq. Ft	Total area of the jets in question.
C	AR	R	I	----	Aspect ratio of the planform.
C	B	R	I	----	Width of current body config.
C	CU	R	I	----	Momentum coef. based on nozzle.
C	DH	R	I	Ft	Vertical position of the jet based on the planform.
C	DI	R	I	Ft	Diameter of the jets in question.
C	FIGWK	C	-	----	Flag which notifies JIEPIE which planform is in use.
C	HEIGHT	R	I	Ft	Height of aircraft
C	KL_F	R	I	----	Adjustment factor used for flap extension 1.0 - Flaps extended .25 - Flaps not extended
C	LT	R	O	----	Lift loss returned by STOLWK
C	LT0	R	-	----	Lift loss out of ground effect.
C	LT_FLP	R	O	----	Basic lift gain due to jet flap action.
C	LT_OG	R	O	----	Out of ground lift loss
C	MC	R	I	----	Mean Aerodynamic Chord.
C	N	I	I	----	Number of nozzles based on which nozzle used (Front, Rear)
C	PR	R	I	----	Pressure Ratio of local jets.
C	Q	R	I	lb/Sq Ft	Jet dynamic pressure
C	Q0	R	O	lb/Sq Ft	Free stream dynamic pressure.
C	S	R	I	Sq Ft	Area of the configuration.
C	SF	R	I	Sq Ft	Area of configuration that is in front of nozzles.
C	T	R	I	lb	Thrust of the jets in question.
C	TEFALG	L	-	----	Flag which notifies STOLWK how close nozzle is to the trailing edge of the wing.
C	WL	R	I	----	Width to length ratio of nozzles.
C	XC	R	-	----	Position of nozzle with respect to wing trailing edge (in % chord)
C	Y	R	I	Ft	Lateral spacing of jets.
C	YP	R	I	Ft	Distance of jet center to bodyside.
C	VEFACT	R	I	----	Effective velocity factor which accounts for reduction in velocity at rear nozzles due to front nozzle
C	Z	R	I		Vertical distance of the nozzles.

C RESPIE Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	A	I	O	----	Angle of attack number counter
C	AOA(500)	R	I	deg	Actual Angle Of Attack value.
C	D	I	O	----	Nozzle deflection angle number counter.
C	DFL(500)	R	I	deg	Actual nozzle deflection angle value
C	GV_LT	R	O	----	Total lift loss due to the ground vortex.

C	GV_LTB	R	O	----	Lift loss due to the ground vortex effect on the body alone
C					
C	GV_LTW	R	O	----	Lift loss due to the ground vortex effect on the wing alone
C					
C	H	I	O	----	Height number counter
C	H_HPRI	R	O	Ft	Height to which tangent line intersects $dL/T=0$ (Used in Hover hprime method, obviously)
C					
C	H_LT(500)	R	O	----	Total lift loss calculated while in hover.
C					
C	H_LTF(500)	R	O	----	Lift gain due to fountain effects while in hover.
C					
C	H_LTL(500)	R	O	----	Lift gain due to the addition of LIDs while in hover.
C					
C	H_LTOG	R	O	----	Lift loss out of ground effect while in hover.
C					
C	HLTOGE(500)	R	O	----	Lift loss out of ground effect while in hover based on deflection angle.
C					
C	H_LTS(500)	R	O	----	Lift loss due to suckdown while in hover.
C					
C	HT(500)	R	O	Ft	Actual height value
C	LA	R	I	----	Total number of Angle of Attack values
C					
C	LD	R	I	----	Total number of nozzle deflection angles.
C					
C	LH	R	I	----	Total number of height values.
C	LV	R	I	----	Total number of velocity values.
C	REC3	I	O	----	Record number for any file which replaces a 3 X 3 matrix
C					
C	REC4	I	O	----	Record number for any file which replaces a 4 X 4 matrix
C					
C	RCS_LT(1,500)	R	O	----	Total lift loss from the Reaction Control System.
C					
C	SF_LT	R	O	----	Total lift/gain due to the suckdown and fountain effects while in forward flight.
C					
C	SF_LTF	R	O	----	Lift gain due to fountain effects while in forward flight.
C					
C	SF_LTS	R	O	----	Lift loss due to suckdown effects while in forward flight
C					
C	V	I	O	----	Velocity number counter
C	VO(500)	R	O	----	Actual velocity value
C	WK_LT	R	O	----	Total lift loss due to the jet wake system produce in forward flight
C					
C	WK_LTB	R	O	----	Lift loss due to jet wake system produced on the body.
C					
C	WK_LTW	R	O	----	Lift loss due to jet wake system produce on the wing.

C FILES USED:

C	LOGICAL UNIT	I/O	DESCRIPTION
C	-----	---	-----
C	68	O	Sequential file for table output
C	70	O	Direct access file for storage of lift increment due to suckdown
C			
C	71	O	Direct access file for storage of lift increment due to fountain
C			

```

C 72      O Direct access file for storage of lift increment
C          due to the ground vortex on the body
C 73      O Direct access file for storage of lift increment
C          due to the ground vortex on the wing
C 74      O Direct access file for storage of lift increment
C          due to the jet wake on the body
C 75      O Direct access file for stroage of lift increment
C          due to the jet wake on the wing (with out center
C          section)
C
C COMMONS USED: (All common blocks used throughout PIE)
C  NAME      DESCRIPTION
C  -----
C  CONPIE    CONTROL variables for Power Induced Effects - Contains
C             variables which control the execution of the PIE module.
C  FIGPIE    conFIGuration for Power Induced Effects - Contains all
C             variables needed to define the configuration and other
C             parameters of the aircraft.
C  HOVPIE    HOVER variables for Power Induced Effects - Contains
C             variables which are sent to the HOV GE subroutine.
C  PIEFLAGS  Power Induced Effects FLAGS - Contains variables which
C             help keep track of the configuration of the aircraft.
C  PIEGV     Power Induced Effects for stolGV - Contains
C             variables which are sent to the STOLGV subroutine.
C  PIERCS    Power Inducde Effects for the Reaction Control System -
C             Contains variables which are sent to the RCSIND
C             subroutine.
C  PIERROR   Power Induced Effects for eRRORS - Contains all error
C             flags within PIE.
C  PIESF     Power Induced Effects for stolSF - Contains
C             variables which are sent to the STOLSF subroutine.
C  PIEWK     Power Induced Effects for stolWK and jiepie - Contains
C             all variables which are passes to the STOLWK and JIEPIE
C             subroutines.
C  POLYGON   Contains the points used in POLYAR when calculating the
C             area of a polygon and CENTAR when calculating the center
C             of area.
C  RESPIE    RESults from all POWER Induced Effects subroutines -
C             Contains results from each subroutine along with
C             variables for height, velocity, jet defelction angle,
C             and angle of attack and their counters.
C
C List of all routines:
C  NAME      DESCRIPTION
C  -----
C  COPPIE    Controls execution and coordination of Power Induced
C             Effects Module
C  INPIE     Sets defaults for variables and reads user defined input
C             file
C  PLANMO    Modifies planforms for consistency and integrity
C  FINVAR    Calculates all variables which have not been defined by
C             the user or set to a default value
C  SDAREA    Calculates areas necessary to make calculations for
C             fountain effect
C  DIABAR    Calculates angular mean diameter of a planform
C  HCALL     Isolates and controls execution of HOV GE
C  HOV_GE    Calculates hover lift increments for OGE and GE
C             suckdown, and fountain effects
C  SFCALL    Isolates and controls execution of STOLSF
C  STOLSF    Calculates suckdown and fountain lift increments while

```

C in STOL flight
 C GVCALL Isolates and controls execution of STOLGV
 C STOLGV Calculates ground vortex lift increments while in STOL
 C flight
 C WKCALL Isolates and controls execution of STOLWK
 C JIEPIE Calculates change in lift caused by the jet induce
 C effects on a flat plate
 C STOLWK Calculates jet wake lift increments while in STOL flight
 C RCSCAL Isolates and controls execution of RCSIND
 C RCSIND Calculates RCS lift increments while in forward flight
 C PIEP Printing routine for PIE
 C JIETAB Creates output file which is used by ACSYNT
 C CENTAR Calculates the center of area, area and area in front of
 C a point for a given planform
 C CROSSIN Calculates the intersection of two lines, with each
 C line defined by two points
 C DIST (FUNC) Calculates the distance between two points
 C INTEGRA Calculates the areas of thin rectangles
 C LINECRO Calculates the intersection of two lines, each defined
 C by a slope and Y-intercept
 C LINTERP Linear interpolates between two X and Y values give an
 C intermediate X value
 C PERPDIS Calculates the perpendicular distance between a point
 C and a line
 C POLYAR Calculates the area of any polygon
 C RTOTAL Sums the total lift increment given array type indices
 C SSEN Calculates start, stop, end, and number variables which
 C have not been defined

C ROUTINES CALLED:

NAME	DESCRIPTION
INPIE	Sets defaults for variables and reads user defined input file
PLANMO	Modifies planforms for consistency and integrity
FINVAR	Calculates all variables which have not been defined by the user or set to a default value
HCALL	Isolates and controls execution of HOV_GE
RCSCAL	Isolates and controls execution of RCSIND
SFCALL	Isolates and controls execution of STOLSF
WKCALL	Isolates and controls execution of STOLWK
GVCALL	Isolates and controls execution of STOLGV
PIEP	Printing routine for PIE

C NOTES: None.

C REFERENCES: (Used throughout the PIE routines)

- 1) Kuhn, R.E. "An Engineering Method of Estimating the Induced Lift on V/STOL Aircraft Hovering In and Out of Ground Effect." V/STOL Consultant. NADC-80246-60. pp. January, 1981.
- 2) Henderson, C., Clark, J., Walters, M. "V/STOL Aerodynamics and Stability & Control Manual" Naval Air Development Center, Pennsylvania, January 15, 1980. NADC-80017-60.
- 3) Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the Aerodynamic Stability and Control Parameters of STOL Aircraft Configurations." North American Aircraft Operations. Rockwell International Corporation. AFWAL-TR-87-3019. Volume II & III. June 1987.

C ENVIRONMENT:

C FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.

C NON-STANDARD CODE:

```

C   ?.
C  AUTHOR(S):
C    Kipp Edward Howard (KEH), Cal Poly San Luis Obispo, CA
C                                     NASA Ames Research Center, Moffet Field CA
C
C  REVISION HISTORY:
C    DATE      INITIALS & DESCRIPTION
C    04/11/90  KEH -- Initial completion of code.
C    07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C-----
C  #include "figpie.inc"
C  #include "pieflag.inc"
C  #include "conpie.inc"
C  #include "respie.inc"
C
C      INTEGER ICALC
C  When ICALC is set to 1 then call the variable input subroutine, INPIE
C  Set ICALC = 1 for testing purposes
C      ICALC = 1
C
C      IF (ICALC .EQ. 1) THEN
C          CALL INPIE
C      END IF
C
C  Open necessary input, output, and scratch files
C  Table Output File
C      OPEN (UNIT=68,STATUS='UNKNOWN')
C  Forward velocity suckdown term
C      OPEN (UNIT=70,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  Forward velocity fountain term
C      OPEN (UNIT=71,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  Ground vortex for the Body
C      OPEN (UNIT=72,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  Ground vortex for the Wing
C      OPEN (UNIT=73,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  Jet wake for the Body
C      OPEN (UNIT=74,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  Jet wake for the Wing
C      OPEN (UNIT=75,FORM='FORMATTED',
C          $   ACCESS='DIRECT',RECL=11,STATUS='SCRATCH')
C  The meat of the Power Induced Effects module.
C  Set ICALC = 2 for testing purposes
C      ICALC = 2
C
C      IF (ICALC .EQ. 2) THEN
C          Make planform modifications
C          CALL PLANMO
C          Finish calculating all initial variables
C          CALL FINVAR
C
C          Start do loops
C          Altitude (Ft)
C          DO H = 1, LH

```

```

C      HT(H) = HSTART + (H - 1) * HSTEP
C      Lift loss in hover
C      CALL HCALL
C
C      Velocity (kts)
C      DO V = 1, LV
C          VO(V) = VSTART + (V - 1) * VSTEP
C          IF (H .EQ. 1) CALL RCSCAL
C          Jet Deflection Angle (Deg)
C
C          DO D = 1, LD
C
C              DFL(D) = DSTART + (D - 1) * DSTEP
C              STOL Suckdown and Fountain Calcs.
C              CALL SFCALL
C              CALL WKCALL
C
C              Angle of attack (Deg)
C              DO A = 1, LA
C                  AOA(A) = ASTART + (A - 1) * ASTEP
C                  CALL GVCALL
C                  MOCALL not implemented yet.
C                  CALL MOCALL
C              END DO
C          END DO
C      END DO
C
C      END DO
C
C      END DO
C
C      Set H, D, V & A to its last value
C      H = H - 1
C      D = D - 1
C      V = V - 1
C      A = A - 1
C      END IF
C      Call PIEP subroutine to print out all variables.
C      Set ICALC = 3 for testing purposes
C      ICALC = 3
C
C      IF (ICALC .EQ. 3) CALL PIEP
C      Close all scratch files
C      CLOSE (70)
C      CLOSE (71)
C      CLOSE (72)
C      CLOSE (73)
C      CLOSE (74)
C      CLOSE (75)
C
C      END

```

Inpie

SUBROUTINE INPIE

```

C-----
C ACRONYM:  Input routine for Powered Induced Effects.
C      --      -      -      -

```

C PURPOSE: The purpose of INPIE is to input all the variables that the
 C user has placed in the input file with the extension .PIE.
 C LOCAL VARIABLES (in addition to the above parameters):

NAME	TYPE	I/O	UNITS	DESCRIPTION
I	I	-	----	Counter
HEADER	C*8	I	----	Flag which defines which planforms are contained within *.pie flie

C GLOBAL VARIABLES (in addition to the above parameters and local vars):

C CONPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
AEND	R	I	deg	Ending value for angle of attack
ASTART	R	I	deg	Starting value for angle of attack
ASTEP	R	I	deg	Step value for angle of attack
DEND	R	I	deg	Ending value for jet deflection angle
DSTART	R	I	deg	Starting value for jet deflection angle
DSTEP	R	I	deg	Step value for jet deflection angle
HEND	R	I	Ft	Ending value for altitude
HSTART	R	I	Ft	Starting value for altitude
HSTEP	R	I	Ft	Step value for altitude
VEND	R	I	kts	Ending value for aircraft velocity
VSTART	R	I	kts	Starting value for aircraft velocity
VSTEP	R	I	kts	Step value for aricraft velocity

C FIGPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
B_B	R	I	Ft	Width of Body
B_CS	R	I	Ft	Width of Center Section
B_JP	R	I	Ft	Width of Jet Pattern
B_W	R	I	Ft	Width of Wing (Wing span)
B_WB	R	I	Ft	Width of Wing-Body
CONFIG	C*18	I	----	Short title of current config
D_F	R	I	Ft	Diameter of each Front jet
D_R	R	I	Ft	Diameter of each Rear jet
D_RCS	R	I	Ft	Diameter of Roll RCS nozzle
DB_B	R	I	Ft	Dbar of Body
DB_CS	R	I	Ft	Dbar of Center Section
DB_W	R	I	Ft	Dbar of Wing
DB_WB	R	I	Ft	Dbar of Wing-Body
DE_F	R	I	Ft	Effective Diameter of Front jets
DE_FR	R	I	Ft	Effective Diameter of Front & Rear jets combined
DE_R	R	I	Ft	Effective Diameter of Rear jets
DR	R	I	----	Density Ratio (Jet / Atm)
E_1	R	I	Ft	Half distance between adjacent jets (1)
E_3	R	I	Ft	Half distance between adjacent jets (3)
E_4	R	I	Ft	Half distance between adjacent jets (4)
F_NAME	C*50	I	----	Name of file which contains body & wing planform coordinates

C	KB	R	I	----	Boundary layer factor
C	KLF	R	I	----	Adjustment factor for flap extension when calculating the lift loss due to jet induced effects.
C					
C					Body contour factor
C	KR	R	I	----	Length of body
C	L_B	R	I	Ft	Length of Center Section
C	L_CS	R	I	Ft	Length of Front jet
C	L_F	R	I	Ft	Length of Rear jet
C	L_R	R	I	Ft	Length of Wing-Body
C	L_WB	R	I	Ft	Mean Aerodynamic Chord of wing
C	MAC	R	I	Ft	Mass flow rate for one RCS nozzle
C	MD_RCS	R	I	lbm/s	Number of data points for Body planform
C	N_BODY	I	I	----	Number of data points for Wing planform
C	N_WING	I	I	----	Number of data points for Wing-Body planform
C	N_WB	I	I	----	Number of divisions to be used when calculating suckdown areas (SDAREA) and Dbars (DIABAR)
C	NDIV	I	I	----	Total NUMBER of jets
C					NUMBER of Front jets
C	NUM	I	I	Ft	NUMBER of Rear jets
C	NUM_F	I	I	----	Total perimeter of jets
C	NUM_R	I	I	----	Ratio of perimeter enclosed by lids to total perimeterC
C	PER_FR	R	I	Ft	Jet Pressure Ratio for front jets
C	PP_LID	R	I	----	Jet Pressure Ratio for all jets
C					Jet Pressure Ratio for rear jets
C	PR_F	R	I	----	Roll RCS Pressure Ratio
C	PR_FR	R	I	----	Total pressure for one roll RCS jet
C	PR_R	R	I	----	Dynamic pressure for the front jets
C	PR_RCS	R	I	----	Dynamic pressure for Front and Rear jets
C	PT_RCS	R	I	lb/sq ft	Dynamic pressure for the rear jets
C	Q_F	R	I	lb/sq ft	Dynamic pressure for roll RCS jets
C	Q_FR	R	I	lb/sq ft	Corner radius of body sides.
C					Area of Body
C	Q_R	R	I	lb/sq ft	Area of Center Section
C	Q_RCS	R	I	lb/sq ft	Area of Front jets
C	R_B	R	I	Ft	Area affected by fountain arm (1)
C	S_B	R	I	Sq Ft	Area affected by fountain arm (3)
C	S_CS	R	I	Sq Ft	Area affected by fountain arm (4)
C	S_F	R	I	Sq Ft	Total jet exit area
C	S_FA1	R	I	Sq Ft	Area enclosed by Jet Pattern
C	S_FA3	R	I	Sq Ft	Area enclosed by LIDS
C	S_FA4	R	I	Sq Ft	Area of Rear jets
C	S_FR	R	I	Sq Ft	Area of Wing
C	S_JP	R	I	Sq Ft	Area of Wing-Body
C	S_LID	R	I	Sq Ft	Actual scale factor which adjusts area of fountain arm (3) to account for the fact that curvature of aircraft's nose tends not to hold fountain arm very well causing area that fountain arm affects to be scaled down.
C	S_R	R	I	Sq Ft	
C	S_W	R	I	Sq Ft	
C	S_WB	R	I	Sq Ft	
C	SCALE	R	I	----	Fountain arm (3) scaler (Percentage
C					
C					
C					
C					
C					
C					
C	SCALE3	R	I	----	

C					measured from the front jets to the
C					nose of the aircraft.) Tries to
C					approximate SCALE but does not do a
C					great job (better than nothing.)
C	SF_FB	R	I	Sq Ft	Area ahead of Front jets using body
C					planform
C	SF_FW	R	I	Sq Ft	Area ahead of Front jets using wing
C					planform
C	SF_FWB	R	I	Sq Ft	Area ahead of Front jets using the
C					wingbody planform
C	SF_RB	R	I	Sq Ft	Area ahead of Rear jets using the
C					body planform
C	SF_RW	R	I	Sq Ft	Area ahead of Rear jets using the
C					wing planform
C	SF_RWB	R	I	Sq Ft	Area ahead of Rear jets using the
C					wingbody planform
C	SP_FA1	R	I	Sq Ft	Potential area affected by fountain
C					arm (1)
C	SP_FA3	R	I	Sq Ft	Potential area affected by fountain
C					arm (3)
C	SP_FA4	R	I	Sq Ft	Potential area affected by fountain
C					arm (4)
C	SP_JP	R	I	Sq Ft	Actual surface area within area
C					enclosed by nozzles
C	SPLY_F	R	I	Ft	SPLaY angle of Front jet
C	SPLY_R	R	I	Ft	SPLaY angle of Rear jet
C	T_F	R	I	lb	Thrust of Front jets
C	T_FR	R	I	lb	Total thrust in all jets
C	T_R	R	I	lb	Thrust of Rear jets
C	T_RCS	R	I	lb	Thrust of roll RCS nozzle
C	THA_1	R	I	deg	Half angle between jets (1)
C	THA_3	R	I	deg	Half angle between jets (3)
C	THA_4	R	I	deg	Half angle between jets (4)
C	TP_RCS	R	I	Rankine	Temperature of flow in roll RCS
C					nozzle
C	TT_F	R	I	----	Front Thrust / total Thrust
C	TT_R	R	I	----	Rear Thrust / total Thrust
C	W_F	R	I	Ft	Width of Front jet
C	W_R	R	I	Ft	Width of Rear jets
C	WIWE	R	I	----	Flow Weight of Inlet / Flow Weight
C					of Exit
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_CS	R	I	----	Width to Length ratio of Center
C					Section
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_JP	R	I	----	Width to Length ratio of Jet
C					Pattern
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	WL_WB	R	I	----	Width to Length ratio of Wing-Body
C	X_BODY(500)	R	I	Ft	X-coordinates of Body planform
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	X_RCS	R	I	Ft	Distance of roll RCS nozzle ahead
C					of wing trailing edge
C	X_WB(500)	R	I	Ft	X-coordinates of Wing-Body planform
C	X_WING(500)	R	I	Ft	X-coordinates of Wing planform
C	XCA_CG	R	I	Ft	Distance of center of area ahead of
C					CG
C	X_CA	R	I	Ft	X-coordinate of Center of Area

C	X.CG	R	I	Ft	X-coordinate of Center of Gravity
C	XCG_C2	R	I	Ft	Distance from CG to MAC/2
C	XCG_C4	R	I	Ft	Distance from CG to MAC/4
C	XCG_F	R	I	Ft	Distance Front jet is ahead of CG
C	XCG_I	R	I	Ft	Inlet longitudinal distance ahead of CG
C					Distance Rear jet is ahead of CG
C	XCG_R	R	I	Ft	X-coordinates of Front NOzzle
C	XNOZ_F	R	I	Ft	X-coordinates of Rear NOzzle
C	XNOZ_R	R	I	Ft	X-coordinate of trailing edge for front nozzle (root TE if Nozzle within body)
C	XTE_F	R	I	Ft	X-coordinate of trailing edge for rear nozzle (root TE if Nozzle within body)
C	XTE_R	R	I	Ft	Spanwise extent of planform on fountain arm (1) center line.
C					Spanwise extent of planform on fountain arm (3) center line.
C	Y_1	R	I	Ft	Spanwise extent of planform on fountain arm (4) center line.
C	Y_3	R	I	Ft	Y-coordinates of Body planform
C	Y_4	R	I	Ft	Distance between Front jets
C	Y_BODY(500)	R	I	Ft	Y-coordinates of Front NOzzle
C	Y_F	R	I	Ft	Y-coordinates of R NOzzle
C	YNOZ_F	R	I	Ft	Distance between Rear jets
C	YNOZ_R	R	I	Ft	Distance of roll RCS nozzle in from wingtip
C	Y_R	R	I	Ft	Y-coordinates of Wing-Body planform
C	Y_RCS	R	I	Ft	Y-coordinates of Wing planform
C	Y_WB(500)	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	Y_WING(500)	R	I	Ft	Lateral distance from Body to Rear jets (for external jets)
C	YB_F	R	I	Ft	Max spanwise extent of planform (1)
C	YB_R	R	I	Ft	Max spanwise extent of planform (3)
C					Max spanwise extent of planform (4)
C	YP_1	R	I	Ft	Lateral distance from Wingbody to Front jets (for external jets)
C	YP_3	R	I	Ft	Lateral distance from wingbody to Rear jets (for external jets)
C	YP_4	R	I	Ft	Height of body base above nozzle
C	YWB_F	R	I	Ft	Inlet vertical distance above CG
C	YWB_R	R	I	Ft	height of wing above nozzle
C	Z_B	R	I	Ft	
C	ZCG_I	R	I	Ft	
C	Z_W	R	I	Ft	
C	-----				
C	PIEFLAG Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	-----	-----
C	HDEOUT	L	I	----	Signals when to output tables based on height or height/De
C					(TRUE - Print based on Height/DE, FALSE - Print based on height)
C	VEOUT	L	I	----	Signals when to output tables based on VE
C					TRUE - Print Based on VE
C					FALSE - Print Based on Velocity
C	WBFLAG	L	O	----	Identifies when WingBody planform has been enter by the user. Used

to determine when to use the
wingbody incalculations.

```

C -----
C RESPIE Common Block
C  NAME      TYPE  I/O  UNITS      DESCRIPTION
C -----
C  LA         R    I    ----      Total number of Angle of Attack v
C                                     values
C  LD         R    I    ----      Total number of nozzle deflection
C                                     angles values.
C  LH         R    I    ----      Total number of height values.
C  LV         R    I    ----      Total number of velocity values.
C FLAG VARIABLES (in addition to the above parameters and local vars):
C  NAME      TYPE  DESCRIPTION
C -----
C
C FILES USED:
C  LOGICAL UNIT  I/O  DESCRIPTION
C -----
C  9              I    Data file containing planform coordinates of body
C                    & wing separately or wingbody coordinates only.
C  10             I    Data file containing namelist PIENAM variables.
C COMMONS USED:
C  NAME      DESCRIPTION
C -----
C  CONPIE     CONTROL variables for Power Induced Effects - Contains
C                    variables which control the execution of the PIE module.
C  FIGPIE     conFIGuration for Power Induced Effects - Contains all
C                    variables needed to define the configuration and other
C                    parameters of the aircraft.
C  PIEFLAGS   Power Induced Effects FLAGS - Contains variables which
C                    help keep track of the configuration of the aircraft.
C  RESPIE     RESults from all POWER Induced Effects subroutines -
C                    Contains results from each subroutine along with
C                    variables for height, velocity, jet defelction angle,
C                    and angle of attack and their counters.
C CALLED BY:
C  NAME      DESCRIPTION
C -----
C  COPPIE     Controls execution and coordination of PIE
C ROUTINES CALLED:
C  NAME      DESCRIPTION
C -----
C  (none)
C NOTES: None.
C REFERENCES:
C  1) None.
C ENVIRONMENT:
C  FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C  ?
C AUTHOR(S):
C  Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C  DATE      INITIALS & DESCRIPTION
C  04/03/90  KEH -- Initial coding complete.
C  07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----

```

```

#include "conpie.inc"
#include "figpie.inc"
#include "pieflag.inc"
#include "respie.inc"
C
C Variable Declarations
CHARACTER HEADER*8
C
C Define namelist PIENAME
NAMELIST/PIENAM/B_B, DB_B, KR, L_B, R_B, S_B, WL_B, Z_B,
$ B_W, DB_W, MAC, S_W, Z_W,
$ B_WB, DB_WB, L_WB, S_WB, WL_WB,
$ B_JP, PP_LID, S_JP, S_LID, SP_JP, WL_JP,
$ B_CS, DB_CS, L_CS, S_CS, WL_CS,
$ D_F, DE_F, L_F, NUM_F, PR_F, Q_F, S_F, SF_FB,
$ SF_FWB, SPLY_F, T_F, TT_F, W_F, WL_F, XCG_F, XNOZ_F,
$ Y_F, YB_F, YNOZ_F, XTE_F, YWB_F,
$ D_R, DE_R, L_R, NUM_R, PR_R, Q_R, S_R, SF_RB,
$ SF_RWB, SPLY_R, T_R, TT_R, W_R, WL_R, XCG_R, XNOZ_R,
$ Y_R, YB_R, YNOZ_R, XTE_R, YWB_R,
$ DE_FR, NUM, PER_FR, Q_FR, S_FR, T_FR, X_FR,
$ DR, KLF, PR_FR, WIWE, XCG_I, ZCG_I, CONFIG,
$ D_RCS, MD_RCS, PR_RCS, PT_RCS, T_RCS,
$ TP_RCS, Q_RCS, X_RCS, Y_RCS,
$ KB, NDIV, X_CG, XCA_CG, XCG_C2, XCG_C4,
$ F_NAME, HDEOUT, VEOUT,
$ S_FA1, S_FA3, S_FA4, SP_FA1, SP_FA3, SP_FA4, SCALE3, SCALE,
$ E_1, E_3, E_4, THA_1, THA_3, THA_4, Y_1, Y_3, Y_4,
$ YP_1, YP_3, YP_4,
$ HSTART, HEND, HSTEP, LH, VSTART, VEND, VSTEP, LV,
$ DSTART, DEND, DSTEP, LD, ASTART, AEND, ASTEP, LA, SKIP
C
C Initialization of all variables:
C Variable that has been set to 9999. is a variable which will be
C calculated if the user does not set it through the input file.
AEND = 9999.
ASTART = 9999.
ASTEP = 9999.
B_B = 9999.
B_CS = 9999.
B_JP = 9999.
B_W = 9999.
B_WB = 9999.
CONFIG = '????????????????????'
D_F = 9999.
D_R = 9999.
D_RCS = 9999.
DB_B = 9999.
DB_CS = 9999.
DB_W = 9999.
DB_WB = 9999.
DE_F = 9999.
DE_FR = 9999.
DE_R = 9999.
DR = 1.0
DEND = 9999.
DSTART = 9999.
DSTEP = 9999.

```

```
E_1 = 9999.  
E_3 = 9999.  
E_4 = 9999.  
F_NAME = 'test.pie'  
HDEOUT = .TRUE.  
HEND = 9999.  
HSTART = 9999.  
HSTEP = 9999.  
KB = .666667  
KLF = 1.0  
KR = 9999.  
L_B = 9999.  
L_CS = 9999.  
L_F = 9999.  
L_R = 9999.  
L_WB = 9999.  
LH = 9999  
LV = 9999  
LD = 9999  
LA = 9999  
MAC = 0.0  
MD_RCS = 0.0  
NDIV = 500  
NUM = 9999  
NUM_F = 9999  
NUM_R = 9999  
PER_FR = 9999.  
PP_LID = 9999.  
PR_F = 9999.  
PR_FR = 9999.  
PR_R = 9999.  
PR_RCS = 0.0  
PRTF LG = .TRUE.  
PT_RCS = 9999.  
Q_F = 9999.  
Q_R = 9999.  
Q_FR = 9999.  
Q_RCS = 9999.  
R_B = 9999.  
S_B = 9999.  
S_CS = 9999.  
S_JP = 9999.  
S_F = 9999.  
S_FA1 = 9999.  
S_FA3 = 9999.  
S_FA4 = 9999.  
S_FR = 9999.  
S_LID = 0.0  
S_R = 9999.  
S_W = 9999.  
S_WB = 9999.  
SCALE = 9999.  
SCALE3 = 1.0  
SF_FB = 9999.  
SF_FWB = 9999.  
SF_RB = 9999.  
SF_RWB = 9999.  
SKIP = .FALSE.
```

```
SP_JP = 9999.
SP_FA1 = 9999.
SP_FA3 = 9999.
SP_FA4 = 9999.
SPLY_F = 0.0
SPLY_R = 0.0
T_F = 9999.
T_FR = 9999.
T_R = 9999.
T_RCS = 0.0
THA_1 = 9999.
THA_3 = 9999.
THA_4 = 9999.
TP_RCS = 0.0
TT_F = 9999.
TT_R = 9999.
VEND = 9999.
VEOUT = .FALSE.
VSTART = 9999.
VSTEP = 9999.
W_F = 9999.
W_R = 9999.
WBFLAG = .FALSE.
WIWE = 0.0
WL_B = 9999.
WL_CS = 9999.
WL_JP = 9999.
WL_F = 9999.
WL_R = 9999.
WL_WB = 9999.
X_FR = 9999.
X_RCS = 9999.
XCA_CG = 9999.
X_CG = 0.0
X_CA = 9999.
XCG_C2 = 9999.
XCG_C4 = 0.0
XCG_F = 9999.
XCG_I = 0.0
XCG_R = 9999.
XNOZ_F = 0.0
XNOZ_R = 9999.
XTE_F = 9999.
XTE_R = 9999.
Y_1 = 9999.
Y_3 = 9999.
Y_4 = 9999.
YP_1 = 9999.
YP_3 = 9999.
YP_4 = 9999.
YNOZ_F = 0.0
YNOZ_R = 9999.
Y_F = 9999.
Y_R = 9999.
Y_RCS = 9999.
YB_F = 9999.
YB_R = 9999.
YWB_F = 9999.
```

```

YWB_R = 9999.
Z_B = 0.0
ZCG_I = 0.0
Z_W = 0.0
C
C Read user entered data in PIENAM Name list
  READ (9,NML=PIENAM)
C Read in planform coordinates from F_NAME. Iris is case sensitive
C when it comes to naming files.
C   Opening file
  OPEN(UNIT = 10, FILE = F_NAME, TYPE = 'OLD')
C   Set up loop to read planform coordinates
5   I = 1
C   Read header for planform coordinates.
  READ (10,7,END=50) HEADER
7   FORMAT (A8)
C
  IF (HEADER .EQ. 'BODYPLAN') THEN
    WBFLAG = .FALSE.
    C   Read body coordinates
    10  READ (10,*) X_BODY(I), Y_BODY(I)
    C   Upon reaching the flag (9999.) for X_BODY, go to next planform.
    C   IF (X_BODY(I) .EQ. 9999.) GO TO 40
    C   Keep track of the number of body points
    N_BODY = I
    C   Increment counter
    I = I + 1
    GO TO 10
  ELSE IF (HEADER .EQ. 'WINGPLAN') THEN
    WBFLAG = .FALSE.
    C   Read wing coordinates
    20  READ (10,*) X_WING(I), Y_WING(I)
    C   Upon reaching the flag (9999.) for X_WING, go to next planform.
    C   IF (X_WING(I) .EQ. 9999.) GO TO 40
    C   Keep track of the number of wing points
    N_WING = I
    C   Increment counter
    I = I + 1
    GO TO 20
  ELSE IF (HEADER .EQ. 'WINGBODY') THEN
    WBFLAG = .TRUE.
    C   Read wingbody coordinates
    30  READ (10,*) X_WB(I), Y_WB(I)
    C   Upon reaching the flag (9999.) for X_WB, go to next planform.
    C   IF (X_WB(I) .EQ. 9999.) GO TO 40
    C   Keep track of the number of wingbody points
    N_WB = I
    C   Increment counter
    I = I + 1
    GO TO 30
  40  END IF
C
  GO TO 5
C
C   Assign the correct value to WBFLAG. WBFLAG is true if only the
C   wingbody is entered alone.
50  CONTINUE
C

```

RETURN
END

Planmo

SUBROUTINE PLANMO

```

C-----
C ACRONYM:  PLANform Modifications.
C-----
C
C PURPOSE:  The purpose of PLANMO is to
C           1) Adjust any two planform coordinates that have the
C              same X-coordinate. (This is because the slope of a
C              line between two points with the same X-coordinate is
C              infinity.)
C           2) Combine the individual body and wing planforms into
C              one planform for the wingbody.
C           3) Move the most forward part of the body planform
C              (generally the nose) to the origin of the coordinate
C              system and move the wing planform likewise.
C
C LOCAL VARIABLES:
C
C   NAME          TYPE  I/O  UNITS  DESCRIPTION
C-----
C   BP(20)        I     -    ----  Body point counter which keep track
C                                     of which body point a crossing has
C                                     has taken place before.
C   CROSS(20)     I     -    ----  Flag which identifies when a
C                                     crossing point has been reached.
C   DELTAX        R           Ft    Amount to add to a point that has
C                                     the same X-coordinate as the
C                                     previous point. (= .1% of the total
C                                     length of the configuration)
C   I              I           ----  Counter for body
C   J              I           ----  Counter for wing
C   K              I           ----  Counter for wingbody
C   L              I           ----  Counter for center section
C   M              I           ----  Crossing counter
C   W_FLAG        I     I    ----  Flag which signifies when to use
C                                     the wing planform when creating
C                                     the wingbody planform.
C                                     1 - use wing planform
C                                     -1 - use body planform
C   WP(20)        I     -    ----  Wing point counter which keep track
C                                     of which wing point a crossing has
C                                     has taken place before.
C   XCR(20)       R     -    ----  X-coordinate at which M crossing
C                                     occurs
C   XCROSS        R           Ft    The X-coordinate of the inter-
C                                     section of two lines defined by two
C                                     body points and two wing points
C   YCR(20)       R     -    ----  Y-coordinate at which M crossing
C                                     occurs
C   YCROSS        R           Ft    See XCROSS (Y-coordinate)
C   XMAX          R           Ft    Most aft point of configuration
C   XMIN          R           Ft    Most forward point of the
C                                     configuration.
C
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C-----
C FIGPIE Common Block

```


NAME	TYPE	I/O	UNITS	DESCRIPTION
N_BODY	I	I	----	Number of data points for Body planform
N_CS	I	O	----	Number of data points for Center Section planform
N_WING	I	I	----	Number of data points for Wing planform
N_WB	I	I	----	Number of data points for Wing-Body planform
X_BODY(500)	R	I	Ft	X-coordinates of Body planform
X_CS(30)	R	O	Ft	X-coordinates of Center Section planform
X_RCS	R	I	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
X_WB(500)	R	I	Ft	X-coordinates of Wing-Body planform
X_WING(500)	R	I	Ft	X-coordinates of Wing planform
X_CA	R	I	Ft	X-coordinate of Center of Area
X_CG	R	I	Ft	X-coordinate of Center of Gravity
XNOZ_F	R	I	Ft	X-coordinates of Front NOzzle
XNOZ_R	R	I	Ft	X-coordinates of Rear NOzzle
XTE_F	R	I	Ft	X-coordinate of trailing edge for front nozzle (root TE if Nozzle within body)
XTE_R	R	I	Ft	X-coordinate of trailing edge for rear nozzle (root TE if Nozzle within body)
Y_CS(30)	R	O	Ft	Y-coordinates of Center Section

PIEFLAG Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
WBFLAG	L	O	----	Identifies when WingBody planform has been enter by the user. Used to determine when to use the wingbody in calculations.

FILES USED:

LOGICAL UNIT	I/O	DESCRIPTION
(none)		

COMMONS USED:

NAME	DESCRIPTION
FIGPIE	conFIGuration for Power Induced Effects - Contains all variables needed to define the configuration and other parameters of the aircraft.
PIEFLAGS	Power Induced Effects FLAGS - Contains variables which help keep track of the configuration of the aircraft.

CALLED BY:

NAME	DESCRIPTION
COPPIE	Controls execution and coordination of PIE

ROUTINES CALLED:

NAME	DESCRIPTION
CROSSING	Calculates the intersection of two lines defined by four points. Line one is defined with (X1,Y1) & (X2,Y2).

```

C          Line two is defined with (X3,Y3) & (X4,Y4).
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   03/15/90  KEH -- Initial coding complete.
C   07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "figpie.inc"
C #include "pieflag.inc"
C
C Variable decalarations
C   REAL CROSS(20), XCR(20), YCR(20)
C   INTEGER W_FLAG, BP(20), WP(20), I, J, K, L, M
C
C ADJUST TWO PLANFORM COORDINATES IF THEY HAVE THE SAME X-COORDINATE
C Find the maximum and minimum points on the planform
C Use the wingbody planform if WBFLAG has been set
C   IF (WBFLAG) THEN
C     DO I = 1, N_WB
C       Initialize XMAX and XMIN to the first points of planform.
C
C       IF (I .EQ. 1) THEN
C         XMIN = X_WB(I)
C         XMAX = X_WB(I)
C       END IF
C
C       Assign wingbody planform coordinate to XMIN if the wingbody
C       planform coordinate is less than XMIN.
C       IF (X_WB(I) .LT. XMIN) XMIN = X_WB(I)
C       Assign wingbody planform coordinate to XMAX if the wingbody
C       planform coordinate is greater than XMAX.
C       IF (X_WB(I) .GT. XMAX) XMAX = X_WB(I)
C     END DO
C   Else use body planform
C   ELSE
C     DO I = 1, N_BODY
C       Initialize XMAX and XMIN to the first points of planform.
C
C       IF (I .EQ. 1) THEN
C         XMIN = X_BODY(I)
C         XMAX = X_BODY(I)
C       END IF
C
C       Assign body planform coordinate to XMIN if the body
C       planform coordinate is less than XMIN.
C       IF (X_BODY(I) .LT. XMIN) XMIN = X_BODY(I)
C       Assign body planform coordinate to XMAX if the body
C       planform coordinate is greater than XMAX.
C       IF (X_BODY(I) .GT. XMAX) XMAX = X_BODY(I)
C     END DO

```

```

      END IF
C
C   Calculate a relative delta X to adjust the given points
C   DELTAX = .001 * (XMAX - XMIN)
C   Use the wingbody planform if WBFLAG is set
C
      IF (WBFLAG) THEN
        DO 30 I = 2, N_WB
          Adjust the current point if the previous point has the
          same X-coordinate.
          IF (X_WB(I-1) .EQ. X_WB(I))
            X_WB(I) = X_WB(I) + DELTAX
30      CONTINUE
C   Else use body and wing planform
      ELSE
        Body adjustment
        DO 40 I = 2, N_BODY
          Adjust the current point if the previous point has the
          same X-coordinate.
          IF (X_BODY(I-1) .EQ. X_BODY(I))
            X_BODY(I) = X_BODY(I) + DELTAX
40      CONTINUE
C   Wing adjustment
        DO 50 I = 2, N_WING
          same X-coordinate.
          IF (X_WING(I-1) .EQ. X_WING(I))
            X_WING(I) = X_WING(I) + DELTAX
50      CONTINUE
      END IF
C-----
C   COMBINE BODY AND WING INTO WINGBODY PLANFORM AND FIND THE
C   CENTER SCETION PLANFORM.
C
      IF (.NOT. WBFLAG) THEN
C   Initialize counters
C   Wingbody counter
        K = 2
C   Crossing counter
        M = 1
C   Determine where the wing planform intersects the body planform
C   and store those points in an array
C   Step through the body planform
        DO I = 2, N_BODY
          Step through the wing planform
          DO J = 2, N_WING
            Check to see if any of the points on the body and wing
            coincide.
C
C
C
            IF (X_BODY(I) .NE. X_WING(J) .AND.
              X_BODY(I-1) .NE. X_WING(J-1)) THEN
C   Find intersection of line defined by wing points and
C   line defined by body points.
              CALL CROSSIN(X_BODY(I-1), Y_BODY(I-1),
                X_BODY(I), Y_BODY(I),
                X_WING(J-1), Y_WING(J-1),
                X_WING(J), Y_WING(J),
                XCROSS, YCROSS)
            $
            $
            $
            $
          ELSE

```

```

        XCROSS = X_BODY(I)
        YCROSS = Y_BODY(I)
    END IF

C
C      Check to see if lines cross
C      IF ((XCROSS .GT. X_BODY(I-1) .AND. XCROSS.LE.X_BODY(I))
$      .OR. (XCROSS .LT.X_BODY(I-1) .AND. XCROSS.GE.X_BODY(I)))
$      .AND. ((XCROSS.GT.X_WING(J-1) .AND. XCROSS.LE.X_WING(J))
$      .OR. (XCROSS .LT.X_WING(J-1) .AND. XCROSS.GE.X_WING(J)))
$      THEN

C
C      Check to see if one of the crossing points is at the
C      same point as a body or wing point.
C      IF (XCROSS .EQ. X_BODY(I) .OR.
$      XCROSS .EQ. X_WING(I)) THEN
        CROSS(M) = 1
      ELSE
        CROSS(M) = -1
      END IF

C
        BP(M) = I
        WP(M) = J
        XCR(M) = XCROSS
        YCR(M) = YCROSS
        M = M + 1
      END IF

C
      END DO
END DO

C      Set end condition for wing and body planform
BP(M) = 0
WP(M) = 0

C      Initialize Body counter
I = 2

C      Initialize Wing counter
J = 2

C      Initialize Wingbody counter
K = 2

C      Initialize Center section counter
M = 1

C
C      Calculate wingbody planform
C      Determine which planform to start with.
C      IF (X_BODY(1) .EQ. 0.0) THEN
C          The body planform
C          X_WB(1) = X_BODY(1)
C          W_FLAG = -1
C      ELSE
C          The wing planform
C          X_WB(1) = X_WING(1)
C          W_FLAG = 1
C      END IF

C      Termination condition - occurs when the wingbody point equals
C      the last point of either the body or wing.
70  IF ((X_WB(K-1) .EQ. X_BODY(N_BODY) .AND.
$      Y_WB(K-1) .EQ. Y_BODY(N_BODY)) .OR.
$      (X_WB(K-1) .EQ. X_WING(N_WING) .AND.
$      Y_WB(K-1) .EQ. Y_WING(N_WING))) GO TO 80

```

```

C      IF (W_FLAG .EQ. -1) THEN
C          Use the body points for the wingbody planform
C
C          IF (I .LT. BP(M) .OR. BP(M) .EQ. 0) THEN
C              X_WB(K) = X_BODY(I)
C              Y_WB(K) = Y_BODY(I)
C              Increment body counter
C              I = I + 1
C              Increment wingbody counter
C              K = K + 1
C              Run through tests again
C              GO TO 70
C          A crossing point has been reached so use it in the wingbody
C          planform.
C          ELSE IF (I .EQ. BP(M)) THEN
C              X_WB(K) = XCR(M)
C              Y_WB(K) = YCR(M)
C              Increment wingbody counter
C              K = K + 1
C
C              Determine counter for wing
C              IF (CROSS(M) .EQ. 1) THEN
C                  J = WP(M) + 1
C              ELSE IF (CROSS(M) .EQ. -1) THEN
C                  J = WP(M)
C              END IF
C
C              Caculate W_FLAG
C              W_FLAG = W_FLAG * -1
C              Increment crossing counter
C              M = M + 1
C              Run through tests again
C              GO TO 70
C          END IF
C
C      ELSE IF (W_FLAG .EQ. 1) THEN
C          Use the wing points for the wingbody planform
C          IF (J .LT. WP(M) .OR. WP(M) .EQ. 0) THEN
C              X_WB(K) = X_WING(J)
C              Y_WB(K) = Y_WING(J)
C              Increment wing counter
C              J = J + 1
C              Increment wingbody counter
C              K = K + 1
C              Run through tests again
C              GO TO 70
C          A crossing point has been reached so use it in the wingbody
C          planform.
C          ELSE IF (J .EQ. WP(M)) THEN
C              X_WB(K) = XCR(M)
C              Y_WB(K) = YCR(M)
C              Increment wingbody counter
C              K = K + 1
C
C              Determine counter for body
C              IF (CROSS(M) .EQ. 1) THEN

```

```

      I = BP(M) + 1
      ELSE IF (CROSS(M) .EQ. -1) THEN
        I = BP(M)
      END IF

C      Caculate W_FLAG
C      W_FLAG = W_FLAG * -1
C      Increment Crossing counter
C      M = M + 1
C      Run through tests again
      GO TO 70
    END IF

C
  END IF

C
C      Finish wingbody calculations
80  N_WB = K - 1
C      Calculate center section planform
C      Initialize counters
      I = 2
      J = 2
      L = 2
      K = 2
      M = 1

C
C      Determine which planform to start with.
      IF (X_BODY(1) .GT. X_WING(1)) THEN
C        Body planform
        X_CS(1) = X_BODY(1)
        W_FLAG = -1
      ELSE
C        Wing planform
        X_CS(1) = X_WING(1)
        W_FLAG = 1
      END IF

C
C      Termination condition - occurs when the center section point
C      equals the last point of either the body or wing.
90  IF ((X_CS(L-1) .EQ. X_BODY(N_BODY) .AND.
$      Y_CS(L-1) .EQ. Y_BODY(N_BODY)) .OR.
$      (X_CS(L-1) .EQ. X_WING(N_WING) .AND.
$      Y_CS(L-1) .EQ. Y_WING(N_WING))) GO TO 100

C
      IF (W_FLAG .EQ. -1) THEN
C        Use the body points for the center section planform
C
        IF (I .LT. BP(M) .OR. BP(M) .EQ. 0) THEN
          X_CS(L) = X_BODY(I)
          Y_CS(L) = Y_BODY(I)
C          Increment body counter
          I = I + 1
C          Increment center section counter
          L = L + 1
C          Run through tests again
          GO TO 90
C          A crossing point has been reached so use it in the wingbody
C          planform.
        ELSE IF (I .EQ. BP(M)) THEN

```

```

      X_CS(L) = XCR(M)
      Y_CS(L) = YCR(M)
C      Increment center section counter
      L = L + 1
C      Determine counter for wing
C
      IF (CROSS(M) .EQ. 1) THEN
        J = WP(M) + 1
      ELSE IF (CROSS(M) .EQ. -1) THEN
        J = WP(M)
      END IF
C
C      Caculate W_FLAG
      W_FLAG = W_FLAG * -1
C      Increment crossing counter
      M = M + 1
C      Run through tests again
      GO TO 90
      END IF
C
      ELSE IF (W_FLAG .EQ. 1) THEN
C
C      Use the wing points for the center section planform.
      IF (J .LT. WP(M) .OR. WP(M) .EQ. 0) THEN
        X_CS(L) = X_WING(J)
        Y_CS(L) = Y_WING(J)
C      Increment wing counter
        J = J + 1
C      Increment center section counter
        L = L + 1
C      Run through tests again
        GO TO 90
C      A crossing point has been reached so use it in the wingbody
C      planform.
      ELSE IF (J .EQ. WP(M)) THEN
        X_CS(L) = XCR(M)
        Y_CS(L) = YCR(M)
C      Increment center section counter
        L = L + 1
C      Determine counter for body
C
        IF (CROSS(M) .EQ. 1) THEN
          I = BP(M) + 1
        ELSE IF (CROSS(M) .EQ. -1) THEN
          I = BP(M)
        END IF
C
C      Caculate W_FLAG
      W_FLAG = W_FLAG * -1
C      Increment crossing counter
      M = M + 1
C      Run through tests again
      GO TO 90
      END IF
C
      END IF
C
      Finish calculations for the center section planform.

```


C	DASTAR	R	-	deg	Default value for starting value of angles of attack
C	DLA	R	-	----	Default value for number of angles of attack
C	DDEND	R	-	deg	Default value for ending value of deflection angle
C	DDSTAR	R	-	deg	Default value for starting value of deflection angle
C	DLD	R	-	----	Default value for number of deflection angles
C	DHEND	R	-	deg	Default value for ending value of heights
C	DHSTAR	R	-	deg	Default value for starting value of heights
C	DLH	R	-	----	Default value for number of heights
C	DVEND	R	-	deg	Default value for ending value of velocities
C	DVSTAR	R	-	deg	Default value for starting value of velocities
C	DLV	R	-	----	Default value for number of velocities
C	N_DB	I	-	----	Number of coordinates passed to DIABAR routine.
C	PER	R	-	Ft	Used to keep track of perimeter when calculating PER_FR
C	PI	R	-	----	Constant: 3.1415926
C	V_RCS	R	-	Ft/S	Velocity of flow in roll RCS nozzle
C	X_DB(500)	R	-	----	X-coordinate for array which is passed to DIABAR routine.
C	XMAX_B	R	-	----	Maximum X-coordinate value for body.
C	XMAXCS	R	-	----	Maximum X-coordinate value for the center section.
C	XMAXWB	R	-	----	Maximum value of the X-coordinates for WINGBODY planform.
C	XMINCS	R	-	----	Minimum X-coordinate value for center section.
C	XMINWB	R	-	----	Minimum value of the X-coordinates for WINGBODY planform.
C	Y_DB(500)	R	-	----	Y-coordinate for array which is passed to DIABAR routine.
C	YB	R	-	Ft	Body width at a particular nozzle location
C	YMAX_B	R	-	----	Maximum Y-coordinate value for the body.
C	YMAXCS	R	-	----	Maximum Y-coordinate value for the center section.
C	YMAXWB	R	-	----	Maximum value of the Y-coordinates for the WINGBODY planform.

C GLOBAL VARIABLES (in addition to the above parameters and local vars):

C -----
C CONPIE Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	-----	-----
C	AEND	R	I	deg	Ending value for angle of attack
C	ASTART	R	I	deg	Starting value for angle of attack
C	ASTEP	R	I	deg	Step value for angle of attack

C	DEND	R	I	deg	Ending value for jet deflection angle
C					
C	DSTART	R	I	deg	Starting value for jet deflection angle
C					
C	DSTEP	R	I	deg	Step value for jet deflection angle
C	HEND	R	I	Ft	Ending value for altitude
C	HSTART	R	I	Ft	Starting value for altitude
C	HSTEP	R	I	Ft	Step value for altitude
C	VEND	R	I	kts	Ending value for aircraft velocity
C	VSTART	R	I	kts	Starting value for aircraft velocity
C					
C	VSTEP	R	I	kts	Step value for aircraft velocity
C					
C	FIGP Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C					
C	B_B	R	I	Ft	Width of Body
C	B_CS	R	I	Ft	Width of Center Section
C	B_JP	R	I	Ft	Width of Jet Pattern
C	B_W	R	I	Ft	Width of Wing (Wing span)
C	B_WB	R	I	Ft	Width of Wing-Body
C	D_F	R	I	Ft	Diameter of each Front jet
C	D_R	R	I	Ft	Diameter of each Rear jet
C	D_RCS	R	I	Ft	Diameter of Roll RCS nozzle
C	DB_B	R	I	Ft	Dbar of Body
C	DB_CS	R	I	Ft	Dbar of Center Section
C	DB_W	R	I	Ft	Dbar of Wing
C	DB_WB	R	I	Ft	Dbar of Wing-Body
C	DE_F	R	I	Ft	Effective Diameter of Front jets
C	DE_FR	R	I	Ft	Effective Diameter of Front & Rear jets combined
C					
C	DE_R	R	I	Ft	Effective Diameter of Rear jets
C	E_1	R	I	Ft	Half distance between adjacent jets (1)
C					
C	E_3	R	I	Ft	Half distance between adjacent jets (3)
C					
C	E_4	R	I	Ft	Half distance between adjacent jets (4)
C					
C	KR	R	I	----	Body contour factor
C	L_B	R	I	Ft	Length of body
C	L_CS	R	I	Ft	Length of Center Section
C	L_F	R	I	Ft	Length of Front jet
C	L_R	R	I	Ft	Length of Rear jet
C	L_WB	R	I	Ft	Length of Wing-Body
C	MAC	R	I	Ft	Mean Aerodynamic Chord of wing
C	MD_RCS	R	I	lbm/s	Mass flow rate for one RCS nozzle
C	N_BODY	I	I	----	Number of data points for Body planform
C					
C	N_CS	I	O	----	Number of data points for Center Section planform
C					
C	N_WING	I	I	----	Number of data points for Wing planform
C					
C	N_WB	I	I	----	Number of data points for Wing-Body planform
C					
C	NDIV	I	I	----	Number of divisions used when calculating suckdown areas (SDAREA) and Dbars (DIABAR)
C					

C	NUM	I	I	Ft	Total NUMBER of jets
C	NUM_F	I	I	----	NUMBER of Front jets
C	NUM_R	I	I	----	NUMBER of Rear jets
C	PER_FR	R	I	Ft	Total perimeter of jets
C	PP_LID	R	I	----	Ratio of perimeter enclosed by lids to total perimeter
C					
C	PR_F	R	I	----	Jet Pressure Ratio for front jets
C	PR_FR	R	I	----	Jet Pressure Ratio for all jets
C	PR_R	R	I	----	Jet Pressure Ratio for rear jets
C	PR_RCS	R	I	----	Roll RCS Pressure Ratio
C	PT_RCS	R	I	lb/sq ft	Total pressure for one roll RCS jet
C	Q_F	R	I	lb/sq ft	Dynamic pressure for the front jets
C	Q_FR	R	I	lb/sq ft	Dynamic pressure for Front and Rear jets
C					
C	Q_R	R	I	lb/sq ft	Dynamic pressure for the rear jets
C	Q_RCS	R	I	lb/sq ft	Dynamic pressure for roll RCS jets
C	R_B	R	I	Ft	Corner radius of body sides.
C	S_B	R	I	Sq Ft	Area of Body
C	S_CS	R	I	Sq Ft	Area of Center Section
C	S_F	R	I	Sq Ft	Area of Front jets
C	S_FA1	R	I	Sq Ft	Area affected by fountain arm (1)
C	S_FA3	R	I	Sq Ft	Area affected by fountain arm (3)
C	S_FA4	R	I	Sq Ft	Area affected by fountain arm (4)
C	S_FR	R	I	Sq Ft	Total jet exit area
C	S_JP	R	I	Sq Ft	Area enclosed by Jet Pattern
C	S_LID	R	I	Sq Ft	Area enclosed by LIDS
C	S_R	R	I	Sq Ft	Area of Rear jets
C	S_W	R	I	Sq Ft	Area of Wing
C	S_WB	R	I	Sq Ft	Area of Wing-Body
C	SF_FB	R	I	Sq Ft	Area ahead of Front jets using body planform
C					
C	SF_FW	R	I	Sq Ft	Area ahead of Front jets using wing planform
C					
C	SF_FWB	R	I	Sq Ft	Area ahead of Front jets using the wingbody planform
C					
C	SF_RB	R	I	Sq Ft	Area ahead of Rear jets using the body planform
C					
C	SF_RW	R	I	Sq Ft	Area ahead of Rear jets using the wing planform
C					
C	SF_RWB	R	I	Sq Ft	Area ahead of Rear jets using the wingbody planform
C					
C	SP_FA1	R	I	Sq Ft	Potential area affected by fountain arm (1)
C					
C	SP_FA3	R	I	Sq Ft	Potential area affected by fountain arm (3)
C					
C	SP_FA4	R	I	Sq Ft	Potential area affected by fountain arm (4)
C					
C	SP_JP	R	I	Sq Ft	Actual surface area within area enclosed by nozzles
C					
C	T_F	R	I	lb	Thrust of Front jets
C	T_FR	R	I	lb	Total thrust in all jets
C	T_R	R	I	lb	Thrust of Rear jets
C	T_RCS	R	I	lb	Thrust of roll RCS nozzle
C	THA_1	R	I	deg	Half angle between jets (1)
C	THA_3	R	I	deg	Half angle between jets (3)
C	THA_4	R	I	deg	Half angle between jets (4)
C					
C	TP_RCS	R	I	Rankine	Temperature of flow in roll RCS nozzle

C	TT_F	R	I	----	Front Thrust / total Thrust
C	TT_R	R	I	----	Rear Thrust / total Thrust
C	W_F	R	I	Ft	Width of Front jet
C	W_R	R	I	Ft	Width of Rear jets
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_CS	R	I	----	Width to Length ratio of Center Section
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_JP	R	I	----	Width to Length ratio of Jet Pattern
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	WL_WB	R	I	----	Width to Length ratio of Wing-Body
C	X_BODY(500)	R	I	Ft	X-coordinates of Body planform
C	X_CS(30)	R	O	Ft	X-coordinates of Center Section planform
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	X_RCS	R	I	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
C	X_WB(500)	R	I	Ft	X-coordinates of Wing-Body planform
C	X_WING(500)	R	I	Ft	X-coordinates of Wing planform
C	XCA.CG	R	I	Ft	Distance of center of area ahead of CG
C	X_CA	R	I	Ft	X-coordinate of Center of Area
C	X.CG	R	I	Ft	X-coordinate of Center of Gravity
C	XCG_C2	R	I	Ft	Distance from CG to MAC/2
C	XCG_C4	R	I	Ft	Distance from CG to MAC/4
C	XCG_F	R	I	Ft	Distance Front jet is ahead of CG
C	XCG_R	R	I	Ft	Distance Rear jet is ahead of CG
C	XNOZ_F	R	I	Ft	X-coordinates of Front NOZZle
C	XNOZ_R	R	I	Ft	X-coordinates of Rear NOZZle
C	XTE_F	R	I	Ft	X-coordinate of trailing edge for front nozzle (root TE if Nozzle within body)
C	XTE_R	R	I	Ft	X-coordinate of trailing edge for rear nozzle (root TE if Nozzle within body)
C	Y_1	R	I	Ft	Spanwise extent of planform on fountain arm (1) center line.
C	Y_3	R	I	Ft	Spanwise extent of planform on fountain arm (3) center line.
C	Y_4	R	I	Ft	Spanwise extent of planform on fountain arm (4) center line.
C	Y_BODY(500)	R	I	Ft	Y-coordinates of Body planform
C	Y_CS(30)	R	O	Ft	Y-coordinates of Center Section planform
C	Y_F	R	I	Ft	Distance between Front jets
C	YNOZ_F	R	I	Ft	Y-coordinates of Front NOZZle
C	YNOZ_R	R	I	Ft	Y-coordinates of R NOZZle
C	Y_R	R	I	Ft	Distance between Rear jets
C	Y_RCS	R	I	Ft	Distance of roll RCS nozzle in from wingtip
C	Y_WB(500)	R	I	Ft	Y-coordinates of Wing-Body planform
C	Y_WING(500)	R	I	Ft	Y-coordinates of Wing planform
C	YB_F	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	YB_R	R	I	Ft	Lateral distance from Body to Rear jets (for external jets)
C	YP_1	R	I	Ft	Maximum spanwise extent of

C	YP_3	R	I	Ft	planform (1)
C					Maximum spanwise extent of
C	YP_4	R	I	Ft	planform (3)
C					Maximum spanwise extent of
C	YWB_F	R	I	Ft	planform (4)
C					Lateral distance from Wingbody to
C	YWB_R	R	I	Ft	Front jets (for external jets)
C					Lateral distance from wingbody to
C	Z_W	R	I	Ft	Rear jets (for external jets)
C					height of wing above nozzle
C	-----				
C	MISC Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	----	-----
C	POINTS	I	I	----	Number of points which define
C					polygon (not to exceed 500)
C	XPTS(500)	R	I	----	X-coordinates of polygon
C	YPTS(500)	R	I	----	Y-coordinates of polygon
C	TOTAL	R	I	----	Total area enclosed by the polygon.
C	-----				
C	PIEFLAG Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	----	-----
C	FLGRCS	L	O	----	Flag which signals if there is an
C					RCS on the configuration
C					(TRUE - RCS, FALSE - No RCS)
C	TYPE_F	C*4	O	----	Description of type of front nozzle
C					(CIRCular, OVAL, RECTangular)
C	TYPE_R	C*4	O	----	Same as TYPE_F but for rear nozzles
C	WBFLAG	L	O	----	Identifies when WingBody planform
C					has been enter by the user. Used
C					to determine when to use the
C					wingbody in calculations.
C	-----				
C	PIERROR Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	----	-----
C	DBERR	I	O	----	Error flag returned by DIABAR. If
C					IFLAG = 1, then an explanation of
C					the problem is given at the time of
C					the error.
C					0 = no errors.
C					1 = values of dbar & area
C					probably bad.
C					2 = moved x-position of dbar
C					point/nozzle point to
C					geometric center.
C					3 = input NANGLE was
C					inappropriate (set to
C					absolute value or 1000).
C					4 = Points entered in wrong
C					direction, routine aborted.
C					Must enter points from left
C					to right.
C					5 = 1st and last points not on
C					x-axis, routine aborted.
C					6 = the sweeping ray intersected
C					more than 4 points or zero

```

C                                     points; values returned may
C                                     be bad.
C   T_ERR      I      O      ----   Thrust numbers (T_F,T_R,T_FR,TT_F,
C                                     TT_R) were entered in the wrong
C                                     combination
C
C -----
C RESPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   LA        R    I    ----      Total number of Angle of Attack
C                                     values
C   LD        R    I    ----      Total number of nozzle deflection
C                                     angles.
C   LH        R    I    ----      Total number of height values.
C   LV        R    I    ----      Total number of velocity values.
C
C FILES USED:
C   LOGICAL UNIT  I/O  DESCRIPTION
C   -----
C   (none)
C
C COMMONS USED:
C   NAME      DESCRIPTION
C   -----
C   CONPIE     Control variables for Power Induced Effects - Contains
C               variables which control the execution of the PIE module.
C   FIGPIE     conFIGuration for Power Induced Effects - Contains all
C               variables needed to define the configuration and other
C               parameters of the aircraft.
C   PIEFLAGS   Power Induced Effects FLAGS - Contains variables which
C               help keep track of the configuration of the aircraft.
C   PIERROR    Power Induced Effects for eRRORS - Contains all error
C               flags within PIE.
C   POLYGON    Contains the points used in POLYAR when calculating the
C               area of a polygon and CENTAR when calculating the center
C               of area.
C   RESPIE     RESults from all Power Induced Effects subroutines -
C               Contains results from each subroutine along with
C               variables for height, velocity, jet defelction angle,
C               and angle of attack and their counters.
C
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   COPPIE     Main driver routine for the PIE module.
C
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   LINTERP    Linear interpolates between two X and Y values give an
C               intermediate X value
C   CENTAR     Calculates the center of area, area and area in front of
C               a point for a given planform
C   POLYAR     Calculates the area of any polygon
C   SDAREA     Calculates areas necessary to make calculations for the
C               fountain effect
C   DIABAR     Calculates angular mean diameter of a planform
C   SSEN       Calculates start, stop, end, and number variables which
C               have not been defined
C
C NOTES: None.
C REFERENCES:
C   1) None.

```

```

C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   04/16/90  KEH -- Initial coding complete.
C   07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----
C   #include "conpie.inc"
C   #include "figpie.inc"
C   #include "misc.inc"
C   #include "pieflag.inc"
C   #include "pierror.inc"
C   #include "respie.inc"
C
C   REAL PI, X_DB(500), Y_DB(500)
C   INTEGER N_DB, DLA, DLD, DLH, DLV
C
C   PI = 3.1415926
C
C   Determine the number and placement of the nozzles
C   Check front nozzle(s) if necessary
C   IF (NUM_F .EQ. 9999.) THEN
C
C       IF (YNOZ_F .EQ. 0.0) THEN
C           NUM_F = 1
C       ELSE
C           NUM_F = 2
C       END IF
C
C   END IF
C
C   Check rear nozzle(s) if necessary
C   IF (NUM_R .EQ. 9999.) THEN
C
C       IF (YNOZ_R .EQ. 9999.) THEN
C           NUM_R = 0
C       ELSE IF (YNOZ_R .EQ. 0.0) THEN
C           NUM_R = 1
C       ELSE
C           NUM_R = 2
C       END IF
C
C   END IF
C
C   Calculate total number of nozzles if necessary
C   IF (NUM .EQ. 9999.) NUM = NUM_F + NUM_R
C   If there are no rear nozzles, assign the values for rear nozzles
C   equal to the value of the front nozzles (this is because SDAREA
C   needs to have at two nozzles input and for the side-by-side
C   configuration only one nozzle is shown on the planform. SDAREA
C   knows this configuration if both nozzles have the same
C   coordinates.
C

```

```

IF (NUM_R .EQ. 0 .AND. NUM_F .GT. 1) THEN
  XNOZ_R = XNOZ_F
  YNOZ_R = YNOZ_F
END IF

C
C Find the extremes (maximums and minimums) for each planform
C Initialize XMAX and XMIN to the first points of the wingbody
C planform.
XMINWB = X_WB(1)
XMAXWB = X_WB(1)
YMAXWB = Y_WB(1)
DO 10 I = 2, N_WB
  C Assign wingbody planform coordinate to XMINWB if the wingbody
  C planform coordinate is less than XMINWB.
  IF (X_WB(I) .LT. XMINWB) XMINWB = X_WB(I)
  C Assign wingbody planform coordinate to XMAXWB if the wingbody
  C planform coordinate is greater than XMAXWB.
  IF (X_WB(I) .GT. XMAXWB) XMAXWB = X_WB(I)
  C Assign wingbody planform coordinate to YMAXWB if the wingbody
  C planform coordinate is greater than YMAXWB.
  IF (Y_WB(I) .GT. YMAXWB) YMAXWB = Y_WB(I)

  C
  IF (X_WB(I) .GT. XNOZ_F .AND. X_WB(I-1) .LE. XNOZ_F)
    $ THEN
    C Determine the Y-coordinate at the nozzle placement using
    C a linear interpolation method.
    CALL LINTERP(XNOZ_F, X_WB(I), X_WB(I-1), Y_WB(I),
    $ Y_WB(I-1), YB)
    C Calculate distance from body side to jet center.
    IF (YWB_F .EQ. 9999.) YWB_F = YNOZ_F - YB
    END IF

    C
    IF (X_WB(I) .GT. XNOZ_R .AND. X_WB(I-1) .LE. XNOZ_R
    $ .AND. NUM_R .GE. 1) THEN
    C Determine the Y-coordinate at the nozzle placement using
    C a linear interpolation method.
    CALL LINTERP(XNOZ_R, X_WB(I), X_WB(I-1), Y_WB(I),
    $ Y_WB(I-1), YB)
    C Calculate distance from body side to jet center.
    IF (YWB_R .EQ. 9999.) YWB_R = YNOZ_R - YB
    END IF

    C
    10 CONTINUE
  C
  IF (.NOT. WBFLAG) THEN
    C Initialize XMAX and XMIN to the first points of the body
    C planform.
    YMAX_B = Y_BODY(1)
    XMAX_B = X_BODY(1)
    DO 20 I = 2, N_BODY
      C Assign body planform coordinate to YMAX_B if the body
      C planform coordinate is greater than YMAX_B.
      IF (Y_BODY(I) .GT. YMAX_B) YMAX_B = Y_BODY(I)
      C Assign wingbody planform coordinate to XMAXWB if wingbody
      C planform coordinate is greater than XMAXWB.
      IF (X_BODY(I) .GT. XMAX_B) XMAX_B = X_BODY(I)
      C Find distance from body to jets if jets are outside body.
    C
  C

```



```

      IF (X_BODY(I) .GT. XNOZ_F .AND. X_BODY(I-1) .LE. XNOZ_F)
$      THEN
C      Determine the Y-coordinate at the nozzle placement using
C      a linear interpolation method.
      CALL LINTERP(XNOZ_F,X_BODY(I),X_BODY(I-1),Y_BODY(I),
$      Y_BODY(I-1),YB)
C      Calculate distance from body side to jet center.
      IF (YB_F .EQ. 9999.) YB_F = YNOZ_F - YB
      END IF

C      IF (X_BODY(I) .GT. XNOZ_R .AND. X_BODY(I-1) .LE. XNOZ_R
$      .AND. NUM_R .GE. 1) THEN
C      Determine the Y-coordinate at the nozzle placement using
C      a linear interpolation method.
      CALL LINTERP(XNOZ_R,X_BODY(I),X_BODY(I-1),Y_BODY(I),
$      Y_BODY(I-1),YB)
C      Calculate distance from body side to jet center.
      IF (YB_R .EQ. 9999.) YB_R = YNOZ_R - YB
      END IF

C
20      CONTINUE
      END IF

C      Initialize XMAX and XMIN to the first points of the center
C      section planform.
      YMAXCS = Y_CS(1)
      XMAXCS = X_CS(1)
      XMINCS = X_CS(1)
      DO 25 I = 2, N_CS
          IF (X_CS(I) .LT. XMINCS) XMINCS = X_CS(I)
          IF (X_CS(I) .GT. XMAXCS) XMAXCS = X_CS(I)
          IF (Y_CS(I) .GT. YMAXCS) YMAXCS = Y_CS(I)
25      CONTINUE
C-----
C Find dimensions of all pieces (body, wing, wingbody, Jet Pattern,
C center section, front and rear nozzles)
      IF (WBFLAG) THEN
C      Wingbody dimensions
C      Width
      IF (B_WB .EQ. 9999.) B_WB = 2.0 * YMAXWB
C      Length
      IF (L_WB .EQ. 9999.) L_WB = XMAXWB
C      Width to length ratio
      IF (WL_WB .EQ. 9999.) WL_WB = B_WB / L_WB
      B_B = 0.0
      L_B = 0.0
      WL_B = 0.0
      B_W = 0.0
      B_CS = 0.0
      L_CS = 0.0
      WL_CS = 0.0
      ELSE
C      Body dimensions
C      Width
      IF (B_B .EQ. 9999.) B_B = 2.0 * YMAX_B
C      Length
      IF (L_B .EQ. 9999.) L_B = XMAX_B
C      Width to length ratio

```

```

      IF (WL_B .EQ. 9999.) WL_B = B_B / L_B
C      Wing dimensions
C      Width
      IF (B_W .EQ. 9999.) B_W = 2.0 * YMAXWB
C      Wingbody dimensions
C      Width
      IF (B_WB .EQ. 9999.) B_WB = 2.0 * YMAXWB
C      Length
      IF (L_WB .EQ. 9999.) L_WB = XMAXWB
C      Width to length ratio
      IF (WL_WB .EQ. 9999.) WL_WB = B_WB / L_WB
C      Center Section
C      Width
      IF (B_CS .EQ. 9999.) B_CS = 2.0 * YMAXCS
C      Length
      IF (L_CS .EQ. 9999.) L_CS = XMAXCS - XMINCS
C      Width to Length ratio
      IF (WL_CS .EQ. 9999.) WL_CS = B_CS / L_CS
      END IF
C
C      Jet Pattern dimensions
C      Width
      IF (B_JP .EQ. 9999 .AND. NUM .GE. 3) THEN
C      Front nozzle is wider than the rear nozzle
      IF (YNOZ_F .GE. YNOZ_R) B_JP = 2 * YNOZ_F
C      Rear nozzle is wider than the front nozzle
      IF (YNOZ_R .GT. YNOZ_F) B_JP = 2 * YNOZ_R
      ELSE IF (B_JP .EQ. 9999) THEN
        B_JP = 0
      END IF
C
C      Width to length ratio
      IF (WL_JP .EQ. 9999 .AND. NUM .GE. 3) THEN
        WL_JP = B_JP / (XNOZ_R - XNOZ_F)
      ELSE IF (WL_JP .EQ. 9999.) THEN
        WL_JP = 0.0
      END IF
C
C      Front nozzles
C      Determine what type of nozzles
      IF (D_F .NE. 9999. .AND. ((W_F .EQ. 9999. .AND. L_F .EQ. 9999.)
$ .OR. (W_F .EQ. L_F))) THEN
        TYPE_F = 'CIRC'
      ELSE IF (D_F .EQ. 9999. .AND. W_F .NE. 9999. .AND. L_F .NE. 9999.)
$ THEN
        TYPE_F = 'RECT'
      ELSE IF (D_F .NE. 9999. .AND. W_F .NE. 9999. .AND. L_F .NE. 9999.)
$ THEN
        TYPE_F = 'OVAL'
      END IF
C
C      Width to length ratio
      IF (TYPE_F .EQ. 'CIRC') THEN
        W_F = D_F
        L_F = D_F
        WL_F = 1.0
      ELSE
        WL_F = W_F / L_F

```

```

END IF

C
C Distance between front jets
IF (NUM_F .GT. 1 .AND. Y_F .EQ. 9999.) THEN
    Y_F = 2.0 * YNOZ_F
ELSE IF (NUM_F .EQ. 1 .AND. Y_F .EQ. 9999.) THEN
    Y_F = 0.0
END IF

C
C Equivalent diameter
IF (DE_F .EQ. 9999.) THEN
    C For circular nozzles
    IF (TYPE_F .EQ. 'CIRC') THEN
        DE_F = SQRT(NUM_F * D_F**2.0)
    C For oval nozzles
    ELSE IF (TYPE_F .EQ. 'OVAL') THEN
        DE_F = SQRT(NUM_F * W_F * L_F)
    C For rectangular nozzles
    ELSE IF (TYPE_F .EQ. 'RECT') THEN
        DE_F = SQRT(4.0/PI*NUM_F*W_F*L_F)
    END IF

C
END IF

C
C Diameter of individual jets if jets are not circular
IF (D_F .EQ. 9999.) D_F = DE_F/SQRT(REAL(NUM_F))

C
C Calculate distance from CG to front nozzle
IF (XCG_F .EQ. 9999.) XCG_F = X.CG - XNOZ_F

C
C Rear nozzles
C Rear nozzles present?
IF (NUM_R .NE. 0) THEN
    C Determine what type of nozzles
    C
    IF (D_R .NE. 9999. .AND. ((W_R .EQ. 9999. .AND. L_R .EQ. 9999.)
    $ .OR. (W_R .EQ. L_R))) THEN
        TYPE_R = 'CIRC'
    ELSE IF (D_R .EQ. 9999. .AND. W_R .NE. 9999. .AND.
    $ L_R .NE. 9999.) THEN
        TYPE_R = 'RECT'
    ELSE IF (D_R .NE. 9999. .AND. W_R .NE. 9999. .AND.
    $ L_R .NE. 9999.) THEN
        TYPE_R = 'OVAL'
    END IF

C
C Width to length ratio
IF (TYPE_R .EQ. 'CIRC') THEN
    W_R = D_R
    L_R = D_R
    WL_R = 1.0
ELSE
    WL_R = W_R / L_R
END IF

C
C Distance between rear jets
IF (NUM_R .GT. 0 .AND. YNOZ_R .NE. 0.0 .AND. Y_R .EQ. 9999.)

```

```

$   THEN
    Y_R = 2.0 * YNOZ_R
  ELSE
    Y_R = 0.0
  END IF

C   Equivalent diameter
C   IF (DE_R .EQ. 9999. .AND. NUM_R .GT. 0) THEN
C       For circular nozzles
C       IF (TYPE_R .EQ. 'CIRC') DE_R = SQRT(NUM_R * D_R**2.0)
C       For oval nozzles
C       IF (TYPE_R .EQ. 'OVAL') DE_R = SQRT(NUM_R * W_R * L_R)
C       For rectangular nozzles
C       IF (TYPE_R .EQ. 'RECT') DE_R = SQRT(4.0/PI*NUM_R*W_R*L_R)
    END IF

C   Diameter of individual jets if jets are not circular
C   IF (D_R .EQ. 9999.) D_R = DE_R/SQRT(REAL(NUM_R))
C   Calculate distance from CG to rear nozzle
C   IF (XCG_R .EQ. 9999. .AND. NUM_R .GE. 1) XCG_R = X_CG - XNOZ_R
C   No Rear nozzles
  ELSE
C       Set all pertinent values to 0.0
        D_R = 0.0
        W_R = 0.0
        L_R = 0.0
        WL_R = 0.0
        DE_R = 0.0
        XCG_R = 0.0
        S_R = 0.0
        Y_R = 0.0
        YB_R = 0.0
        SF_RB = 0.0
        SF_RW = 0.0
        SF_RWB = 0.0
        W_R = 0.0
        TYPE_R = 'NONE'
    END IF

C   Both nozzles
C   Distance between front and rear nozzles
C   IF (NUM_R .GT. 0 .AND. X_FR .EQ. 9999.) THEN
        X_FR = XNOZ_R - XNOZ_F
    ELSE IF (X_FR .EQ. 9999.) THEN
        X_FR = 0.0
    END IF

C   Equivalent diameter
C   IF (DE_FR .EQ. 9999.0) THEN
C       IF (NUM_R .GT. 0) THEN
            DE_FR = SQRT(DE_F**2.0 + DE_R**2.0)
        ELSE
            DE_FR = DE_F
        END IF
    END IF

C   END IF
C

```

```

C      Total perimeter of jets
      IF (PER_FR .EQ. 9999.) THEN
C
C      Front jets
      IF (TYPE_F .EQ. 'CIRC') THEN
        PER = PI * D_F
      ELSE IF (TYPE_F .EQ. 'OVAL') THEN
        PER = 2.0 * PI * SQRT((W_F**2.0 + L_F**2.0) / 2.0)
      ELSE IF (TYPE_F .EQ. 'RECT') THEN
        PER = 2.0 * W_F + NUM_F * L_F
      END IF
C
      PER_FR = NUM_F * PER
C
C      Rear jets
      IF (NUM_R .GT. 0) THEN
C
        IF (TYPE_R .EQ. 'CIRC') THEN
          PER = PI * D_R
        ELSE IF (TYPE_R .EQ. 'OVAL') THEN
          PER = 2.0 * PI * SQRT((W_R**2.0 + L_R**2.0) / 2.0)
        ELSE IF (TYPE_R .EQ. 'RECT') THEN
          PER = 2.0 * W_R + 2.0 * L_R
        END IF
C
        PER_FR = PER_FR + NUM_R * PER
      END IF
C
      END IF
C-----
C Find values dealing with areas for planforms (DBAR, CENTER OF AREA,
C AREA AHEAD AND BEHIND JETS, ETC.)
C Find the center of area for the wingbody configuration
      IF (X_CA .EQ. 9999.) CALL CENTAR(1, 9999., X_WB, Y_WB, N_WB, X_CA)
C Calculate XCA_CG
      IF (XCA_CG .EQ. 9999.) XCA_CG = X_CG - X_CA
C
C Find total area of body
      IF (S_B .EQ. 9999.) THEN
C
        IF (.NOT. WBFLAG) THEN
          CALL CENTAR(2, 9999., X_BODY, Y_BODY, N_BODY, S_B)
C          The area has to be multiplied by 2 because only half the
C          the planform is defined.
          S_B = 2.0 * S_B
        ELSE
          S_B = 0.0C
        END IF
C
      END IF
C
C Find total area of wing
      IF (S_W .EQ. 9999.) THEN
C
        IF (.NOT. WBFLAG) THEN
          CALL CENTAR(2, 9999., X_WING, Y_WING, N_WING, S_W)
C          The area has to be multiplied by 2 because only half the
C          the planform is defined.

```

```

      S_W = 2.0 * S_W
    ELSE
      S_W = 0.0
    END IF
C
  END IF
C
C Find total area of wingbody
C IF (S_WB .EQ. 9999.) THEN
  CALL CENTAR(2, 9999., X_WB, Y_WB, N_WB, S_WB)
C   The area has to be multiplied by 2 because only half the
C   the planform is defined.
  S_WB = 2.0 * S_WB
  END IF
C
C Find total area of center section
C IF (S_CS .EQ. 9999.) THEN
C   IF (.NOT. WBFLAG) THEN
    CALL CENTAR(2, 9999., X_CS, Y_CS, N_CS, S_CS)
C    The area has to be multiplied by 2 because only half the
C    the planform is defined.
    S_CS = 2.0 * S_CS
  ELSE
    S_CS = 0.0
  END IF
C
  END IF
C
C Find the area ahead of the front nozzle
C Using the body planform
C IF (SF_FB .EQ. 9999.) THEN
C   IF (.NOT. WBFLAG) THEN
    CALL CENTAR(3, XNOZ_F, X_BODY, Y_BODY, N_BODY, SF_FB)
C    The area has to be multiplied by 2 because only half the
C    the planform is defined.
    SF_FB = 2.0 * SF_FB
  ELSE
    SF_FB = 0.0
  END IF
C
  END IF
C
C Using the wing planform
C IF (SF_FW .EQ. 9999.) THEN
C   IF (.NOT. WBFLAG) THEN
    CALL CENTAR(3, XNOZ_F, X_WING, Y_WING, N_WING, SF_FW)
C    The area has to be multiplied by 2 because only half the
C    the planform is defined.
    SF_FW = 2.0 * SF_FW
  ELSE
    SF_FW = 0.0
  END IF
C
  END IF
C

```

```

C      Using the wingbody planform
C      IF (SF_FWB .EQ. 9999.) THEN
C          CALL CENTAR(3, XNOZ_F, X_WB, Y_WB, N_WB, SF_FWB)
C          The area has to be multiplied by 2 because only half the
C          the planform is defined.
C          SF_FWB = 2.0 * SF_FWB
C      END IF

C
C      Find the area ahead of the rear nozzles if necessary.
C      Using the body planform
C      IF (NUM_R .GT. 0) THEN
C
C          IF (SF_RB .EQ. 9999.) THEN
C
C              IF (.NOT. WBFLAG) THEN
C                  CALL CENTAR(3, XNOZ_R, X_BODY, Y_BODY, N_BODY, SF_RB)
C                  The area has to be multiplied by 2 because only half the
C                  the planform is defined.
C                  SF_RB = 2.0 * SF_RB
C              ELSE
C                  SF_RB = 0.0
C              END IF
C
C          END IF
C
C      Using the wing planform
C      IF (SF_RW .EQ. 9999.) THEN
C
C          IF (.NOT. WBFLAG) THEN
C              CALL CENTAR(3, XNOZ_R, X_WING, Y_WING, N_WING, SF_RW)
C              The area has to be multiplied by 2 because only half the
C              the planform is defined.
C              SF_RW = 2.0 * SF_RW
C          ELSE
C              SF_RW = 0.0
C          END IF
C
C      END IF
C
C      Using the wingbody planform
C      IF (SF_RWB .EQ. 9999.) THEN
C          CALL CENTAR(3, XNOZ_R, X_WB, Y_WB, N_WB, SF_RWB)
C          The area has to be multiplied by 2 because only half the
C          the planform is defined.
C          SF_RWB = 2.0 * SF_RWB
C      END IF
C
C      END IF
C
C      Find the area enclosed by the Jet Pattern
C      IF (S_JP .EQ. 9999. .AND. NUM .GE. 3) THEN
C          XPTS(1) = XNOZ_F
C          YPTS(1) = 0.0
C          XPTS(2) = XNOZ_F
C          YPTS(2) = YNOZ_F
C          XPTS(3) = XNOZ_R
C          YPTS(3) = YNOZ_R
C          XPTS(4) = XNOZ_R

```

```

      YPTS(4) = 0.0
      POINTS = 4
      CALL POLYAR
C      The area has to be multiplied by 2 because only half the
C      the planform is defined.
      S_JP = 2.0 * TOTAL
      ELSE IF (S_JP .EQ. 9999.) THEN
        S_JP = 0.0
      END IF

C      Set default value for actual surface area enclosed by jet pattern
C      IF (SP_JP .EQ. 9999. .OR. SP_JP .GT. S_JP) THEN
C
C        IF (NUM .GE. 3) THEN
C          SP_JP = S_JP
C        ELSE
C          SP_JP = 0.0
C        END IF
C
C      END IF

C      Determine the area of each nozzle
C      Front nozzles (using DE because the shape of nozzle has
C      been accounted for when DE was calculated)
C      IF (S_F .EQ. 9999.) S_F = PI / 4.0 * DE_F**2.0
C
C      IF (NUM_R .GT. 0) THEN
C        Rear nozzles (using DE because the shape of nozzle has
C        been accounted for when DE was calculated)
C        IF (S_R .EQ. 9999.) S_R = PI / 4.0 * DE_R**2.0
C        Front and rear nozzles (using DE because the shape of nozzles
C        have been accounted for when DE was calculated)
C        IF (S_FR .EQ. 9999.) S_FR = S_F + S_R
C      ELSE
C        S_R = 0.0
C        S_FR = S_F
C      END IF

C-----
C      Misc. Calculations
C      Jet dynamic pressures
C      IF (PR_FR .NE. 9999.) THEN
C        Assume choked, isentropic flow at nozzle exit, const. Gamma.
C        Qjet = 1/2 * RHOe * Vexit^2
C        RHOe = Pe/(R * Te)
C        Vexit = SQRT(Gamma * R * Te)
C        Po = PR * Patm
C        Using isentropic pressure ratio for choked flow (Mach # = 1),
C        atmospheric pressure, and Pressure Ratio and after reducing
C        all numbers the following equation applies.
C
C        Constants used:      Pe/Po = .52828
C                             R = 53.34 #f*ft/(#m*R)
C                             Gamma = 1.4
C
C        IF (Q_FR .EQ. 9999.) Q_FR = 782.92 * PR_FR
C        Q_F = Q_FR
C        PR_F = PR_FR
C
C      IF (NUM_R .GE. 1) THEN

```



```

      Q_R = Q_FR
      PR_R = PR_FR
    ELSE
      Q_R = 0.0
      PR_R = 0.0
    END IF

C
ELSE IF (PR_F .NE. 9999.) THEN
C
  Same as above.
  IF (Q_F .EQ. 9999.) Q_F = 782.92 * PR_F
C
  IF (PR_R .NE. 9999.) THEN
    Q_R = 782.92 * PR_R
    Q_FR = (Q_F + Q_R) / 2.0
    PR_FR = (PR_F + PR_R) / 2.0
  ELSE
C
    IF (NUM_R .GE. 1) THEN
      Q_R = Q_F
      PR_R = PR_F
    ELSE
      Q_R = 0.0
      PR_R = 0.0
    END IF
C
    Q_FR = Q_F
    PR_FR = PR_F
  END IF
C
END IF

C
Account for different thrust inputs
IF (NUM_R .EQ. 0) THEN
C
  IF (T_F .NE. 9999.) THEN
    T_FR = T_F
  ELSE
    T_F = T_FR
  END IF
C
  TT_F = 1.0
  T_R = 0.0
  TT_R = 0.0
C
  - Input of front and rear thrusts only
  ELSE IF (T_F .NE. 9999. .AND. T_R .NE. 9999.) THEN
    T_FR = T_F + T_R
    TT_F = T_F / T_FR
    TT_R = 1.0 - TT_F
C
  - Input of front thrust split and total thrust.
  ELSE IF (TT_F .NE. 9999. .AND. T_FR .NE. 9999.) THEN
    T_F = T_FR * TT_F
    T_R = T_FR - T_F
    TT_R = T_R / T_FR
C
  - Input of rear thrust split and total thrust.
  ELSE IF (TT_R .NE. 9999. .AND. T_FR .NE. 9999.) THEN
    T_R = T_FR * TT_R
    T_F = T_FR - T_R
    TT_F = T_F / T_FR

```

```

C      - Input of front thrust split and front thrust.
      ELSE IF (TT_F .NE. 9999. .AND. T_F .NE. 9999.) THEN
          T_FR = T_F / TT_F
          T_R = T_FR - T_F
          TT_R = T_R / T_FR
C      - Input of rear thrust split and rear thrust.
      ELSE IF (TT_R .NE. 9999. .AND. T_R .NE. 9999.) THEN
          T_R = T_R / TT_R
          T_F = T_R - T_R
          TT_F = T_F / T_FR
C      - Input of total thrust and rear thrust.
      ELSE IF (T_FR .NE. 9999. .AND. T_R .NE. 9999.) THEN
          T_F = T_FR - T_R
          TT_F = T_F / T_FR
          TT_R = T_R / T_FR
C      - Input of total thrust and front thrust.
      ELSE IF (T_FR .NE. 9999. .AND. T_F .NE. 9999.) THEN
          T_R = T_FR - T_F
          TT_F = T_F / T_FR
          TT_R = T_R / T_FR
C      - Error with thrust inputs
      ELSE
          T_ERR = 1
      END IF

C
C      Calculate the distance from the CG to the mid-span point
C      of the MAC.
      IF (XCG_C2 .EQ. 9999.) XCG_C2 = XCG_C4 - MAC / 4.0

C
C      Set default values for PP_LID based on configuration.
      IF (PP_LID .EQ. 9999. ) THEN
C
          IF (NUM .GE. 3 .AND. S_LID .NE. 0.0) THEN
              PP_LID = 1.0
          ELSE
              PP_LID = 0.0
          END IF

C
      END IF

C
C      Calculate placement of trailing edge of wing relative to the rear
C      nozzle if there are rear nozzles and trailing edge placement
C      has not been entered.
      IF (NUM_R .GT. 0 .AND. XTE_R .EQ. 9999.) XTE_R = XTE_F

C
C      Calculate body contour factor (KR) (NADC-80246-60 Pg 38 Eq. 26)
      IF (KR .EQ. 9999.) THEN
C
          IF (R_B .EQ. 9999.) THEN
              Set default value
              KR = .666667
          ELSE IF (NUM_R .EQ. 0) THEN
              Length wise fountain
              KR = .05 * (R_B / (Y_F/2.0))**-1.0
          ELSE
              Core-and-Arm, and cross wise fountains
              KR = .54 * (R_B / (X_FR/2.0))**-1.20
          END IF
      END IF

```

```

C      END IF
C-----
C Calculate all variables that define fountain arms
C   Body planform
C   IF (Z_W .GT. 0.0 .AND. .NOT. WBFLAG) THEN
C       POINTS = N BODY
C       DO 30 I = 1, POINTS
C           XPTS(I) = X_BODY(I)
C           YPTS(I) = Y_BODY(I)
30      CONTINUE
C   Wingbody planform
C   ELSE
C       POINTS = N WB
C       DO 40 I = 1, POINTS
C           XPTS(I) = X_WB(I)
C           YPTS(I) = Y_WB(I)
40      CONTINUE
C   END IF

C   IF (NUM .GT. 1) THEN
C       CALL SDAREA(0, SDAERR, XPTS, YPTS, POINTS, NDIV)
C   ELSE
C       E_1 = 0.0
C       E_3 = 0.0
C       E_4 = 0.0
C       Y_1 = 0.0
C       Y_3 = 0.0
C       Y_4 = 0.0
C       YP_1 = 0.0
C       YP_3 = 0.0
C       YP_4 = 0.0
C       THA_1 = 0.0
C       THA_3 = 0.0
C       THA_4 = 0.0
C       S_FA1 = 0.0
C       S_FA3 = 0.0
C       S_FA4 = 0.0
C       SP_FA1 = 0.0
C       SP_FA3 = 0.0
C       SP_FA4 = 0.0
C   END IF

C
C Calculate DBAR for Wingbody if only wingbody is given
C   IF (WBFLAG) THEN
C
C       Wingbody planform
C       IF (DB_WB .EQ. 9999.) THEN
C           POINTS = N WB
C           DO I = 1, POINTS
C               XPTS(I) = X_WB(I)
C               YPTS(I) = Y_WB(I)
C           END DO
C           CALL DIABAR(0, DBERR, XPTS, YPTS, POINTS, XNOZ_F+X_FR/2.0,
C               NDIV, DB_WB)
C       END IF
C
C       DB_B = 0.0

```

```

      DB_W = 0.0
      DB_CS = 0.0
ELSE
C
C   Calculate DBAR for each planform
C   Body planform
      IF (DB_B .EQ. 9999.) THEN
        POINTS = N_BODY
        DO 50 I = 1, POINTS
          XPTS(I) = X_BODY(I)
          YPTS(I) = Y_BODY(I)
50      CONTINUE
        CALL DIABAR(0,DBERR, XPTS, YPTS, POINTS, XNOZ_F+X_FR/2.0,
$          NDIV, DB_B)
      END IF
C
C   Wing planform
      IF (DB_W .EQ. 9999.) THEN
        POINTS = N_WING
        DO 60 I = 1, POINTS
          XPTS(I) = X_WING(I)
          YPTS(I) = Y_WING(I)
60      CONTINUE
        CALL DIABAR(0,DBERR, XPTS, YPTS, POINTS, XNOZ_F+X_FR/2.0,
$          NDIV, DB_W)
      END IF
C
C   Wingbody planform
      IF (DB_WB .EQ. 9999.) THEN
        POINTS = N_WB
        DO 70 I = 1, POINTS
          XPTS(I) = X_WB(I)
          YPTS(I) = Y_WB(I)
70      CONTINUE
        CALL DIABAR(0,DBERR, XPTS, YPTS, POINTS, XNOZ_F+X_FR/2.0,
$          NDIV, DB_WB)
      END IF
C
C   Center section planform
      IF (DB_CS .EQ. 9999.) THEN
        POINTS = N_CS
        DO 80 I = 1, POINTS
          XPTS(I) = X_CS(I)
          YPTS(I) = Y_CS(I)
80      CONTINUE
        CALL DIABAR(0,DBERR, XPTS, YPTS, POINTS, XNOZ_F+X_FR/2.0,
$          NDIV, DB_CS)
      END IF
C
      END IF
C-----
C Determine roll RCS variables when necessary.
      IF (X_RCS .EQ. 9999. .OR. Y_RCS .EQ. 9999.) THEN
        FLGRCS = .FALSE.
      ELSE
        FLGRCS = .TRUE.
      END IF
C

```

```

C Determine roll RCS variables when necessary.
  IF (FLGRCS) THEN
C   Find pressure ratio of nozzle
    IF (PR_RCS .EQ. 9999.) then
C
      if (PT_RCS .eq. 9999.) then
C        Assume Q_RCS is known
          PT_RCS = Q_RCS / 144.0
          PR_RCS = PT_RCS / 14.7
        ELSE IF (Q_RCS .EQ. 9999.) THEN
C          Assume PT_RCS is known
            PR_RCS = PT_RCS / 14.7
          END IF
C
        END IF
C
      Find total pressure of the nozzle
      IF (PT_RCS .EQ. 9999.) PT_RCS = PR_RCS * 14.7
C      Find roll jet dynamic pressure
      IF (Q_RCS .EQ. 9999.) Q_RCS = PT_RCS * 144.0
C
      Find roll jet diameter
      IF (D_RCS .EQ. 9999.) THEN
C        Velocity (assume choked flow at the nozzle)
          V_RCS = SQRT(1.4 * 53.34 * 32.2 * TP_RCS)
C        Area of nozzle
          A_RCS = (T_RCS - MD_RCS * V_RCS / 32.2) /
$          (PT_RCS - 14.7) / 144.
C        Diameter of Nozzle
          D_RCS = 2.0 * SQRT(A_RCS / PI)
        END IF
C
      ELSE
        PT_RCS = 0.0
        Q_RCS = 0.0
        D_RCS = 0.0
      END IF
C
C Make looping variable calculations
C   Set default values for looping variables
C   Angle of Attack
    DAEND = 30.
    DASTAR = 0.
    DLA = 4
C   Deflection angle
    DDEND = 30.
    DDSTAR = 90.
    DLD = 7
C   Height
    DHEND = 20 * DE_FR
    DHSTAR = 2 * DE_FR
    DLH = 25
C   Velocity
    DVEND = 100.
    DVSTAR = 1
    DLV = 20
C   Obtain starting, stepping, ending and number (SSEN) values for
C   the following looping variables

```

```

C   Angle of Attack
C   CALL SSEN(ASTART, ASTEP, AEND, LA, DASTAR, DAEND, DLA, 9999.)
C   Deflection Angle
C   CALL SSEN(DSTART, DSTEP, DEND, LD, DDSTAR, DDEND, DLD, 9999.)
C   Height
C   CALL SSEN(HSTART, HSTEP, HEND, LH, DHSTAR, DHEND, DLH, 9999.)
C   Velocity
C   CALL SSEN(VSTART, VSTEP, VEND, LV, DVSTAR, DVEND, DLV, 9999.)
C
C   RETURN
C   END

```

Sdarea

SUBROUTINE SDAREA(IFLAG, SDAERR, X, Y, NPTS, NSLICE)

```

C-----
C PURPOSE:  THIS ROUTINE WILL CALCULATE THE POSITION OF THE FOUNTAIN
C           CORE IN RELATION TO THE NOZZLE POSITION, HALF THE INCLUDED
C           ANGLE BETWEEN THE FOUNTAIN ARMS, HALF THE DISTANCE BETWEEN
C           ADJACENT FOUNTAIN ARMS, SPANWISE EXTENT OF PLANFORM ON
C           FOUNTAIN ARM CENTERLINE, MAXIMUM SPANWISE EXTENT OF
C           PLANFORM BETWEEN JETS, ACTUAL SURFACE AREA AND POTENTIAL
C           SURFACE AREA BETWEEN FOUNTAIN ARMS.  AN INPUT FILE
C           CONTAINING PLANFORM COORDINATES (IN A CLOCKWISE DIRECTION
C           STARTING FROM THE NOSE) IS REQUIRED ALONG WITH THE
C           COORDINATES OF THE NOZZLE.
C
C PARAMETERS:
C   NAME      TYPE  I/O  UNITS  DESCRIPTION
C   -----
C   IFLAG      I    I    ----  USER INTERACTION FLAG:
C                               1 = USER INTERACTION (CAN RUN AS
C                               A STAND ALONE ROUTINE.
C                               OTHERWISE = NO USER INTERACTION
C                               (ALL INPUTS FROM THE CALLING
C                               ROUTINE).
C   SDAERR      O    I    ----  ERROR FLAG RETURNED BY SDAREA.  IF
C                               IFLAG = 1, THEN AN EXPLANATION OF THE
C                               PROBLEM IS GIVEN AT TIME OF THE ERROR.
C                               0 = NO ERRORS.
C                               1 = THERE WAS AT LEAST ONE VERTICAL
C                               LINE WITHIN PLANFORM DATA.
C                               (SECOND VALUE PERTURBED BY .05
C                               UNITS).
C                               2 = NOZZLE HAVE BEEN PLACED EITHER
C                               IN FRONT OF OR BEHIND THE AIRCRAFT
C                               (ROUTINE ABORTED).
C                               3 = INPUT NSLICE WAS INAPPROPRIATE
C                               (SET TO ABSOLUTE VALUE OR 1000).
C                               4 = POINTS ENTERED IN WRONG DIRECTION,
C                               ROUTINE ABORTED. MUST ENTER
C                               POINTS FROM LEFT TO RIGHT.
C                               5 = 1ST AND LAST POINTS NOT ON X-AXIS,
C                               ROUTINE ABORTED.
C                               6 = THE NUMBER OF SLICES THAT WERE
C                               ADDED BY COMPUTER EXCEEDS MAXIMUM
C                               VALUE OF 2000 INTERNAL SLICES.
C                               DECREASE NUMBER OF SLICES.
C   X (500)    R    FEET      X-VALUES OF THE PLANFORM (LIMIT = 500).
C-----

```

VALUES MUST START FROM THE LEFT AND
 PROCEED COUNTERCLOCKWISE FOR ONE HALF OF
 THE AIRCRAFT (THE AIRCRAFT IS ASSUMED
 SYMMETRICAL).

Y (500) R FEET Y-VALUES OF THE PLANFORM (LIMIT = 500).
 NPTS I ---- NUMBER OF POINTS USED TO DEFINE THE
 PLANFORM.

NSLICE I I ---- NUMBER OF SLICES IN THE PLANFORM USED
 IN THE INTEGRATION TECHNIQUE.

LOCAL VARIABLES (IN ADDITION TO THE ABOVE PARAMETERS):

NAME	TYPE	UNITS	DESCRIPTION
A	R	FEET	USED AS UPPER LIMIT FOR INTEGRATION SUBROUTINE (COMPLEMENTARY TO B)
ATYPE	I	----	FLAG WHICH SIGNALS WHICH AREA TO ADD THE AREA FOUND IN A PARTICULAR SLICE.
B	R	FEET	USED AS LOWER LIMIT FOR INTEGRATION SUBROUTINE (COMPLEMENTARY TO A)
BLINE1	R	FEET	Y-INTERCEPT FOR LINE 1
BLINE2	R	FEET	Y-INTERCEPT FOR LINE 2
BLINE3	R	FEET	Y-INTERCEPT FOR LINE 3
BLSPAN	R	FEET	Y-INTERCEPT FOR SPAN LINE (FOUNTAIN LINE)
BMSPAN	R	FEET	Y-INTERCEPT OF LINE WHICH INTERSECTS POINT WHERE MSPAN(1) OCCURS.
BTEST	R	FEET	Y-INTERCEPT USED TO TEST WHERE MSPAN(1) OCCURS.
C	R	FEET	USED AS UPPER LIMIT FOR INTEGRATION SUBROUTINE WHEN A AND B ARE BEING USED (COMPLEMENTARY TO 0.0)
DX	R	FEET	MAXIMUM WIDTH OF EACH SLICE IN THE PLANFORM.
DXX	R	FEET	ACTUAL WIDTH OF EACH SLICE USED IN CALCULATIONS.
E1	R	FEET	HALF THE DISTANCE BETWEEN EACH FOUNTAIN ARM AND ITS NEAREST NOZZLE (IN THE CLOCKWISE DIRECTION, STARTING AT THE SIDE FOUNTAIN ARM.
E2	R	FEET	SEE E1
E3	R	FEET	SEE E1
E4	R	FEET	SEE E1
FACT	R	----	FACTOR WHICH ADDS OR SUBTRACTS AREA ACCORDING TO WHICH DIRECTION THE SLICER IS MOVING.
HOLE(3)	R	----	HOLE DETERMINATION VALUE: 0.0 = THERE IS NO HOLE IN FUSELAGE. 1.0 = THERE IS A HOLE IN THE FUSELAGE. DO NOT INCLUDE THE PARTICULAR FOUNTAIN ARM IN CALCULATIONS.
I	I	----	GENERAL PURPOSE COUNTER
II	I	----	TEMPORARY VALUE OF I
IMAX	I	----	THE POINT NUMBER AT WHICH THE MAXIMUM X VALUE OCCURS
IMIN	I	----	THE POINT NUMBER AT WHICH THE MINIMUM X VALUE OCCURS
LINE1	R	FEET	VALUE OF LINES RADIATING FROM NOZZLE 1, PARALLEL TO THE FOUNTAIN ARM.
LINE2	R	FEET	VALUE OF LINE BETWEEN BOTH NOZZLE
LINE3	R	FEET	VALUE OF LINES RADIATING FROM NOZZLE 2,

C				PARALLEL TO THE FOUNTAIN ARM.
C	LINE4	R	FEET	VALUE OF LINES RADIATING FROM NOZZLE 2,
C				PARALLEL TO THE AIRCRAFT CENTER LINE.
C	LINE5	R	FEET	VALUE OF LINES RADIATING FROM NOZZLE 1,
C				PARALLEL TO THE AIRCRAFT CENTER LINE.
C	LSPAN	R	FEET	VALUE OF LINE THAT REPRESENTS FOUNTAIN
C				ARM.
C	LSPANO	R	FEET	PREVIOUS VALUE OF LSPAN.
C	M1	R	----	SLOPE OF LINE FROM NOZZLE 1 TO FOUNTAIN
C				CORE
C	M2	R	----	SLOPE OF LINE FROM NOZZLE 2 TO FOUNTAIN
C				CORE
C	MLINE1	R	----	SLOPE OF LINE 1
C	MLINE2	R	----	SLOPE OF LINE 2
C	MLINE3	R	----	SLOPE OF LINE 3
C	MLSPAN	R	----	SLOPE OF SPAN LINE (FOUNTAIN ARM).
C	MMSPAN	R	----	SLOPE OF LINE WHICH HELPS DEFINE MSPAN
C				IN REGION 1.
C	MTEMP	R	----	SLOPE OF MIDDLE FOUNTAIN ARM IN THE
C				FOUR NOZZLE CONFIGURATION.
C	MSPAN (3)	R	FEET	MAXIMUM SPANWISE EXTENT OF PLANFORM
C				BETWEEN JETS.
C	N		----	ANOTHER GENERAL PURPOSE COUNTER.
C	NOZARE (2)	R	----	AREA OF NOZZLE WHICH CAN BE SUBTRACTED
C				FROM THE TOTAL AREA FOUND UNDER EACH
C				FOUNTAIN ARM.
C	NOZZLE	I	----	NUMBER OF NOZZLE (MAXIMUM OF 4)
C	PLACE (2)	T	----	TEXT DESCRIBING HOW MANY NOZZLE THERE
C				ARE AT A PARTICULAR PLACE ON PLANFORM.
C	RAD (2)	R	FEET	RADIUS OF EACH NOZZLE
C	RADTEM	R	FEET	TEMPORARY VALUE OF RADIUS USED IN
C				SORTING THE NOZZLE POSITIONS.
C	SAB	R	SQ FT.	AREA BETWEEN 'A' AND 'B' THAT THE
C				INTEGRA SUBROUTINE CALCULATES.
C	SC	R	SQ FT.	AREA BETWEEN 'C' AND 0.0 THAT THE
C				INTEGRA SUBROUTINE CALCULATES.
C	SP1	R	SQ FT.	AREA OF PLANFORM THAT IS AFFECTED BY
C				FOUNTAIN ARM IN REGION 1.
C	SP2	R	SQ FT.	AREA OF PLANFORM THAT IS AFFECTED BY
C				FOUNTAIN ARM IN REGION 2.
C	SP3	R	SQ FT.	AREA OF PLANFORM THAT IS AFFECTED BY
C				FOUNTAIN ARM IN REGION 3.
C	SPP1	R	SQ FT.	POTENTIAL AREA IN REGION 1.
C	SPP2	R	SQ FT.	POTENTIAL AREA IN REGION 2.
C	SPP3	R	SQ FT.	POTENTIAL AREA IN REGION 3.
C	SPAN (3)	R	FEET	SPANWISE EXTENT OF PLANFORM BETWEEN JETS.
C	STOT (3)	R	SQ FT.	TOTAL AREA FOUND UNDER EACH FOUNTAIN ARM.
C	THETA1	R	RADIANS	HALF THE INCLUDED ANGLE BETWEEN ADJACENT
C				FOUNTAIN ARMS.
C	THETA2	R	RADIANS	SEE THETA1
C	THETA3	R	RADIANS	SEE THETA1
C	THETA4	R	RADIANS	SEE THETA1
C	TYPE (2)	T	----	TEXT DESCRIBING WHETHER THE NOZZLE ARE
C				INTERNAL OR EXTERNAL.
C	VERB (2)	T	----	TEXT USED TO ALLOW THE DISCRPTION
C				SENTENCE TO BE GRAMATICALLY CORRECT.
C	X1	R	FEET	1ST X-COORDINATE USED TO DEFINE THE
C				POTENTIAL AREA OF REGION 1.

C	X2	R	FEET	2ND X-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	X3	R	FEET	3RD X-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	X4	R	FEET	4TH X-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	XCORE	R	FEET	X-COORDINATE OF THE FOUNTAIN CORE.	
C	XMAX	R	FEET	MOST AFT POINT ON THE PLANFORM.	
C	XMIN	R	FEET	MOST FORWARD POINT ON THE PLANFORM.	
C	XNOZ (2)	R	FEET	X-COORDINATE OF EACH NOZZLE	
C	XSLICE (2000)	R	FEET	POSITION INDICATOR ACROSS PLANFORM.	
C	XSPAN	R	FEET	X-COORDINATE OF SPANWISE EXTENT OF PLANFORM BETWEEN JETS.	
C	XTEMP	R	FEET	GENERAL PURPOSE TEMPORARY VALUE OF X USED IN MANY DIFFERENT CALCULATIONS.	
C	XYMAX	R	FEET	THE X-COORDINATE THAT CORRESPONDS TO THE Y-COORDINATE THAT IS THE FARTHEST POINT FROM THE AIRCRAFT CENTERLINE (YMAX).	
C	Y1	R	FEET	1ST Y-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	Y2	R	FEET	2ND Y-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	Y3	R	FEET	3RD Y-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	Y4	R	FEET	4TH Y-COORDINATE USED TO DEFINE THE POTENTIAL AREA OF REGION 1.	
C	YCORE	R	FEET	Y-COORDINATE OF THE FOUNTAIN CORE.	
C	YMAX	R	FEET	FARTHEST POINT FROM THE AIRCRAFT CENTER LINE.	
C	YNOZ (2)	R	FEET	Y-COORDINATE OF EACH NOZZLE	
C	YPRIME	R	FEET	VALUE USED TO DETERMINE M2 FOR 3 NOZZLE	
C	YSLICE (2000)	R	FEET	Y-VALUE OF PLANFORM AT THE SLICE POINT.	
C	YSPAN	R	FEET	Y-COORDINATE OF SPANWISE EXTENT OF PLANFORM BETWEEN JETS.	
C	YTEMP	R	FEET	GENERAL PURPOSE TEMPORARY VALUE OF Y USED IN MANY DIFFERENT CALCULATIONS.	
C	YXMAX	R	FEET	THE Y-COORDINATE THAT CORRESPONDS TO THE MOST AFT POINT ON THE PLANFORM (XMAX).	
C	YXMIN	R	FEET	THE Y-COORDINATE THAT CORRESPONDS TO THE MOST FORWARD POINT ON THE PLANFORM (XMIN).	
C	GLOBAL VARIABLES (in addition to the above parameters and local vars):				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	-----	-----
C	ANG_FR	R	I	Ft	Front and Rear jet deflection ANGLE
C	B_B	R	I	Ft	Width of Body
C	B_JP	R	I	Ft	Width of Jet Pattern
C	B_W	R	I	Ft	Width of Wing (Wing span)
C	B_WB	R	I	Ft	Width of Wing-Body
C	D_F	R	I	Ft	Diameter of each Front jet
C	D_R	R	I	Ft	Diameter of each Rear jet
C	D_RCS	R	I	Ft	Diameter of Roll RCS nozzle
C	DB_B	R	I	Ft	Dbar of Body
C	DB_W	R	I	Ft	Dbar of Wing
C	DB_WB	R	I	Ft	Dbar of Wing-Body
C	DE_F	R	I	Ft	Effective Diameter of Front jets
C	DE_FR	R	I	Ft	Effective Diameter of Front & Rear

C					jets combined
C	DE_R	R	I	Ft	Effective Diameter of Rear jets
C	DR	R	I	----	Density Ratio (Jet / Atm)
C	F_NAME	C	I	----	Name of file which contains nozzle
C					placement & size, body & wing
C					planform coordinates
C	H	R	I	FT	Height of nozzle exit above ground
C	ICALC	I	I	----	Global execution flag for ACSYNT
C					1) Input data
C					2) Make calculations
C					3) Output necessary data
C	KB	R	I	----	Boundary layer factor
C	KR	R	I	----	Body contour factor
C	L_B	R	I	Ft	Length of body
C	L_F	R	I	Ft	Length of Front jet
C	L_R	R	I	Ft	Length of Rear jet
C	L_WB	R	I	Ft	Length of Wing-Body
C	MAC	R	I	Ft	Mean Aerodynamic Chord of wing
C	MD_RCS	R	I	lbm/s	Mass flow rate for one RCS nozzle
C	N_BODY	I	I	----	Number of data points for Body
C					planform
C	N_WING	I	I	----	Number of data points for Wing
C					planform
C	N_WB	I	I	----	Number of data points for Wing-Body
C					planform
C	NUM	I	I	Ft	Total NUMBER of jets
C	NUM_F	I	I	----	NUMBER of Front jets
C	NUM_R	I	I	----	NUMBER of Rear jets
C	PER_FR	R	I	Ft	Total perimeter of jets
C	PP_LID	R	I	----	Ratio of perimeter enclosed by lids
C					to total perimeterC
C	PR_FR	R	I	----	Jet Pressure Ratio for all jets
C	PR_RCS	R	I	----	Roll RCS Pressure Ratio
C	PT_RCS	R	I	lb/sq ft	Total pressure for one roll RCS jet
C	Q_FR	R	I	lb/sq ft	Dynamic pressure for Front & Rear
C					jets
C	Q_RCS	R	I	lb/sq ft	Dynamic pressure for roll RCS jets
C	S_B	R	I	Sq Ft	Area of Body
C	S_JP	R	I	Sq Ft	Area enclosed by Jet Pattern
C	S_F	R	I	Sq Ft	Area of Front jets
C	S_FR	R	I	Sq Ft	Total jet exit area
C	S_LID	R	I	Sq Ft	Area enclosed by LIDS
C	S_R	R	I	Sq Ft	Area of Rear jets
C	S_W	R	I	Sq Ft	Area of Wing
C	S_WB	R	I	Sq Ft	Area of Wing-Body
C	SF_FB	R	I	Sq Ft	Area ahead of Front jets using body
C					planform
C	SF_FWB	R	I	Sq Ft	Area ahead of Front jets using
C					wingbody planform
C	SF_RB	R	I	Sq Ft	Area ahead of Rear jets
C	SF_RWB	R	I	Sq Ft	Area ahead of Rear jets using the
C					wingbody planform
C	SP_JP	R	I	Sq Ft	Actual surface area within Jet
C					Pattern
C	SPLY_F	R	I	Ft	SPLaY angle of Front jet
C	SPLY_R	R	I	Ft	SPLaY angle of Rear jet
C	T_F	R	I	lb	Thrust of Front jets
C	T_R	R	I	lb	Thrust of Rear jets

C	T_RCS	R	I	lb	Thrust of roll RCS nozzle
C	TP_RCS	R	I	Rankin	Temperature of flow in roll RCS nozzle
C					Front Thrust / total Thrust
C	TT_F	R	I	----	Rear Thrust / total Thrust
C	TT_R	R	I	----	Forward velocity of aircraft
C	VO	R	I	kts	Width of Front jet
C	W_F	R	I	Ft	Width of Rear jets
C	W_R	R	I	Ft	Flow Weight of Inlet / Flow Weight of Exit
C	WIWE	R	I	----	Width to Length ratio of Body
C					Width to Length ratio of Jet
C	WL_B	R	I	----	Pattern
C	WL_JP	R	I	----	Width to Length ratio of Front jets
C					Width to Length ratio of Rear jets
C	WL_F	R	I	----	Width to Length ratio of Wing-Body
C	WL_R	R	I	----	X-coordinates of Body planform
C	WL_WB	R	I	----	Distance between Front & Rear jets
C	X_BODY(500)	R	I	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
C	X_FR	R	I	Ft	X-coordinates of Wing-Body planform
C	X_RCS	R	I	Ft	X-coordinates of Wing planform
C					Distance of center of area ahead of CG
C	X_WB(500)	R	I	Ft	X-coordinate of Center of Area
C	X_WING(500)	R	I	Ft	X-coordinate of Center of Gravity
C	XCA_CG	R	I	Ft	Distance from CG to MAC/2
C					Distance from CG to MAC/4
C	X_CA	R	I	Ft	Distance Front jet is ahead of CG
C	X_CG	R	I	Ft	Inlet longitudinal distance ahead of CG
C	XCG_C2	R	I	Ft	Distance Rear jet is ahead of CG
C	XCG_C4	R	I	Ft	X-coordinates of Front NOzzle
C	XCG_F	R	I	Ft	X-coordinates of Rear NOzzle
C	XCG_I	R	I	Ft	Y-coordinates of Body planform
C					Distance between Front jets
C	XCG_R	R	I	Ft	Y-coordinates of Front NOzzle
C	XNOZ_F	R	I	Ft	Y-coordinates of R NOzzle
C	XNOZ_R	R	I	Ft	Distance between Rear jets
C	Y_BODY(500)	R	I	Ft	Distance of roll RCS nozzle in from wingtip
C	Y_F	R	I	Ft	Y-coordinates of Wing-Body planform
C	YNOZ_F	R	I	Ft	Y-coordinates of Wing planform
C	YNOZ_R	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	Y_R	R	I	Ft	Lateral distance from Body to Rear jets (for external jets)
C	Y_RCS	R	I	Ft	Height of body base above nozzle
C					Inlet vertical distance above CG
C	Y_WB(500)	R	I	Ft	height of wing above nozzleC
C	Y_WING(500)	R	I	Ft	
C	YB_F	R	I	Ft	
C					
C	YB_R	R	I	Ft	
C					
C	Z_B	R	I	Ft	
C	ZCG_I	R	I	Ft	
C	Z_W	R	I	Ft	
C	COMMONS USED: (All common blocks used throughout PIE)				
C	NAME	DESCRIPTION			
C	-----	-----			
C	CONPIE	CONTROL variables for Power Induced Effects - Contains variables which control the execution of the PIE module.			
C	FIGPIE	conFIGuration for Power Induced Effects - Contains all variables needed to define the configuration and other parameters of the aircraft.			

```

C FILES USED:
C LOGICAL UNIT I/O DESCRIPTION
C -----
C 7 I Input file, used when IFLAG is set to 1.
C NON-STANDARD CODE:
C ?
C NOTES: CAN BE RUN AS A STAND ALONE ROUTINE IF CALLED WITH IFLAG
C SET = 1.
C CALLED BY:
C NAME DESCRIPTION
C -----
C COPPIE Controls execution and coordination of Power Induced
C Effects Module
C SUBROUTINES CALLED:
C NAME DESCRIPTION
C -----
C CROSSIN Calculates the intersection of two lines, with each
C line defined by two points
C DIST (FUNC) Calculates the distance between two points
C INTEGRA Calculates the areas of thin rectangles
C LINECRO Calculates the intersection of two lines, each defined
C by a slope and Y-intercept
C LINTERP Linear interpolates between two X and Y values give an
C intermediate X value
C PERPDIS Calculates the perpendicular distance between a point
C and a line
C
C ENVIRONMENT:
C VAX, VMS, FORTRAN
C
C AUTHOR(S):
C Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C
C REVISION HISTORY:
C DATE INITIALS & DESCRIPTION
C 02/12/90 KEH -- ORIGINAL CODE COMPLETED
C 07/14/91 KEH -- Eliminated extraneous spacing and fixed comments
C -----
#include "figpie.inc"
C
REAL DX,DXX,E1,E2,E3,E4,LINE1,LINE2,LINE3,LINE4,LINE5,M1,M2,
$ RAD(2),RADTEM,THETA1, THETA2, THETA3, THETA4, X(500), XNOZ(2),
$ XMIN, XMAX, YMAX, XTEMP, Y(500), YTEMP, YNOZ(2), XCORE, YCORE,
$ YPRIME, YSLICE(2000), XSLICE(2000), STOT(3), A, B, C, MMSPAN,
$ SPAN(3), MSPAN(3), XSPAN, LSPAN, LSPANO, MLINE1,MLINE2,HOLE(3),
$ MLINE3, MLSPAN, BLINE1, BLINE2, BLINE3, BLSPAN, NOZARE(2)
INTEGER ATYPE, I, II, IFLAG, SDAERR, NSLICE, NOZZLE, NPTS, FACT
CHARACTER USERFI*50, PLACE(2)*15, TYPE(2)*9, VERB(2)*3

C
C Initialization of variables
HOLE(1) = 0.0
HOLE(2) = 0.0
HOLE(3) = 0.0

C
C I/O WITH THE USER (IFLAG = 1).
IF (IFLAG.EQ. 1) THEN
WRITE(6,*)' '
WRITE(6,*)'THIS SUBROUTINE IS DESIGNED TO WORK ONLY FOR

```

```

$ SYMMETRICAL'
  WRITE(6,*) 'AIRPLANES. ALSO THE THRUST IN EACH NOZZLE IS'
  WRITE(6,*) 'ASSUMED TO BE CONSTANT AND THE NOZZLE ARE ASSUMED'
  WRITE(6,*) 'TO BE CIRCULAR.'
  WRITE(6,*) ' '
  WRITE(6,*) 'USE A TEXT FILE TO LIST THE COORDINATES FOR THE
$ NOZZLE FIRST, THEN LIST THE COORDINATES FOR YOUR AIRCRAFT
$ IN A CLOCKWISE DIRECTION SEQUENTIALLY FROM THE LEFT (THE
$ NOSE OF THE AIRCRAFT).'
  WRITE(6,*) ' '
3  WRITE(6,*) ' '
  WRITE(6,*) 'INPUT FILENAME THAT CONTAINS NOZZLE POSITIONS, X
$ & Y COORDINATES:'
  READ(*,5) USERFI
5  FORMAT(A50)
  ENDIF

C
6  IF (IFLAG.EQ. 1) THEN
  WRITE(6,*) ' '
  WRITE(6,*) 'HOW MANY SLICES IN THE PLANFORM WOULD YOU LIKE TO
$USE WHEN DETERMINING THE AREAS? (MAXIMUM = 1000)'
  READ *, NSLICE
  ENDIF

C
  SDAERR = 0
  IF (NSLICE.LT.1) THEN

C
    IF (IFLAG.EQ.0) THEN
      NSLICE = ABS(NSLICE)
      SDAERR = 3

C
      IF (NSLICE.LT.1) THEN
        NSLICE = 1000
      ENDIF

C
    ELSE
      WRITE (6,*) ' '
      WRITE (6,*) 'SORRY DAVE, I CAN NOT DO THAT (HAL).'
      GO TO 6
    ENDIF

C
  ELSE IF (NSLICE.GT.1000) THEN
    NSLICE = 1000
    SDAERR = 3
  ENDIF

C
  READ DATA FROM INPUT FILE (NOZZLE PLACEMENT, NOZZLE RADIUS AND
C  COORDINATES OF PLANFORM. ALSO FIND MOST FORWARD, MOST AFT, AND
C  WING TIP POINTS.
C  IF (IFLAG.EQ. 1) THEN
    OPEN (UNIT = 7, NAME = USERFI, TYPE = 'OLD')
    READ (7,*) XNOZ(1),YNOZ(1),RAD(1),XNOZ(2),YNOZ(2),RAD(2)
    I = 1
10  READ (7, *, END = 12) X(I), Y(I)
    GO TO 10
12  NPTS = I-1
    ELSE
      XNOZ(1) = XNOZ_F

```

```

      YNOZ(1) = YNOZ_F
      XNOZ(2) = XNOZ_R
      YNOZ(2) = YNOZ_R
END IF
C
DO 13 I = 1, NPTS
  IF (I .EQ. 1) THEN
    XMIN = X(I)
    YXMIN = Y(I)
    XMAX = X(I)
    YMAX = Y(I)
    IMIN = I
    IMAX = I
  END IF
  IF ((X(I) .EQ. X(I-1)) .AND. (I .NE. 1)) THEN
    SDAERR = 1
    IF (IFLAG .EQ. 1) THEN
      WRITE (6,*) 'ONE LINE SEGMENT IN THE PLANFORM DATA WAS
$ VERTICAL.'
      WRITE (6,*) '(THE SECOND VALUE WAS MOVED ALONG THE X-AXIS
$ BY .05.)'
    END IF
    X(I) = X(I) + .0001*X(I)
  END IF
  IF (X(I) .LT. XMIN) THEN
    XMIN = X(I)
    YXMIN = Y(I)
    IMIN = I
  END IF
  IF (X(I) .GT. XMAX) THEN
    XMAX = X(I)
    YXMAX = Y(I)
    IMAX = I
  END IF
  IF (Y(I) .GT. YMAX) THEN
    YMAX = Y(I)
    XYMAX = X(I)
  END IF
13 CONTINUE
C MOVE ALL POINTS SO THAT THE MOST FORWARD PART OF THE AIRCRAFT IS
C AT X=0.
C
  IF (XMIN .NE. 0.0) THEN
    DO 15 I = 1, NPTS
      X(I) = X(I) - XMIN
15 CONTINUE
    XNOZ(1) = XNOZ(1) - XMIN
    XNOZ(2) = XNOZ(2) - XMIN
  END IF
C
C SORT NOZZLE SO THAT NOZZLE ONE IS CLOSEST TO THE NOSE OF THE
C AIRCRAFT.
  IF (XNOZ(1) .GT. XNOZ(2)) THEN
    XTEMP = XNOZ(1)
    YTEMP = YNOZ(1)
    RADTEM = RAD(1)
    XNOZ(1) = XNOZ(2)
    YNOZ(1) = YNOZ(2)

```

```

RAD(1) = RAD(2)
XNOZ(2) = XTEMP
YNOZ(2) = YTEMP
RAD(2) = RADTEM
END IF
C
C   CATCHING FAULTY INPUT SO AS TO AVOID INACCURATE DATA IF POSSIBLE.
C   IF (X(1).GT.X(NPTS)) THEN
C
C       IF (IFLAG.EQ.1) THEN
C           WRITE(6,*) 'ERROR IN DATA. PLEASE ENTER COORDINATES FROM
$ LEFT TO RIGHT.'
C           SDAERR = 4
C           GO TO 500
C       ELSE
C           SDAERR = 4
C           GO TO 500
C       ENDIF
C   ENDIF
C
C   IF ((Y(1) .NE. Y(NPTS)) .OR. (Y(1) .NE. 0)) THEN
C
C       IF (IFLAG.EQ.1) THEN
C           WRITE(6,*) 'PLEASE, TRY TO FOLLOW THE DIRECTIONS. HALF AN
$ AIRPLANE ON THE X-AXIS. I.E. BOTH YOUR FIRST AND LAST Y VALUES
$ SHOULD BE 0. THANK YOU.'
C           SDAERR = 5
C           GO TO 500
C       ELSE
C           SDAERR = 5
C           GO TO 500
C       ENDIF
C   ENDIF
C
C   CHECK TO SEE IF NOZZLE ARE IN FRONT OR BEHIND PLANFORM
C   IF ((XNOZ(1).LT.XMIN) .OR. (XNOZ(2).GT.XMAX)) THEN
C
C       IF (IFLAG.EQ.1) THEN
C           WRITE (6,*) 'THE NOZZLES HAVE BEEN LOCATED IN FRONT OR
$ BEHIND THE AIRCRAFT. THE NOZZLE MUST BE PLACED BETWEEN THE
$ NOSE AND THE TAIL OF THE AIRCRAFT.'
C           SDAERR = 2
C           GO TO 500
C       ELSE
C           SDAERR = 2
C           GO TO 500
C       END IF
C   END IF
C
C   IDENTIFY THE NUMBER AND PLACEMENT OF NOZZLES AS A CHECK FOR THE
C   USER.
C   NUMBER OF NOZZLES:
C   IF (((YNOZ(1).EQ.0.0).AND.(YNOZ(2).EQ.0.0)).OR.
$ (XNOZ(1).EQ.XNOZ(2))) THEN

```

```

      NOZZLE = 2
    ELSE IF ((YNOZ(1).GT.0.0) .AND. (YNOZ(2).GT.0.0)) THEN
      NOZZLE = 4
    ELSE
      NOZZLE = 3
    END IF

```

C
C

```

VERBS FOR THE NUMBER OF NOZZLE
IF (NOZZLE.EQ.3) THEN
  IF (YNOZ(1).GT.0.0) THEN
    PLACE(1) = 'ARE TWO NOZZLE'
    VERB(1) = 'ARE'
    PLACE(2) = 'IS ONE NOZZLE '
    VERB(2) = 'IS '
  ELSE
    PLACE(1) = 'IS ONE NOZZLE '
    VERB(1) = 'IS '
    PLACE(2) = 'ARE TWO NOZZLE'
    VERB(2) = 'ARE'
  END IF
ELSE IF (NOZZLE.EQ.2) THEN
  IF (XNOZ(1).EQ.XNOZ(2)) THEN
    PLACE(1) = 'ARE TWO NOZZLE'
    VERB(1) = 'ARE'
  ELSE
    PLACE(1) = 'IS ONE NOZZLE '
    PLACE(2) = PLACE(1)
    VERB(1) = 'IS '
    VERB(2) = VERB(1)
  END IF
ELSE
  PLACE(1) = 'ARE TWO NOZZLE'
  PLACE(2) = PLACE(1)
  VERB(1) = 'ARE'
  VERB(2) = VERB(1)
END IF

```

C
C

```

PLACEMENT OF NOZZLE
DO 20 I=1,NPTS
  IF ((X(I).GE.XNOZ(1)) .AND. (X(I-1).LT.XNOZ(1))) THEN
    CALL LINTERP(XNOZ(1),X(I),X(I-1),Y(I),Y(I-1),ytemp)
    IF (YNOZ(1).GT.YTEMP) THEN
      TYPE(1) = 'EXTERNAL.'
    ELSE
      TYPE(1) = 'INTERNAL.'
    END IF
  END IF
  IF ((X(I).GE.XNOZ(2)) .AND. (X(I-1).LT.XNOZ(2))) THEN
    CALL LINTERP(XNOZ(2),X(I),X(I-1),Y(I),Y(I-1),ytemp)
    IF (YNOZ(2).GT.YTEMP) THEN
      TYPE(2) = 'EXTERNAL.'
    ELSE
      TYPE(2) = 'INTERNAL.'
    END IF
  END IF
END IF

```

C
C

```

DETERMINE WETHER THERE IS A HOLE IN THE FRONT OR REAR OF THE
AIRCRAFT.
  IF (I.LT.IMIN.AND.Y(I).EQ.0.0.AND.Y(I-1).GT.Y(I))

```



```

$   HOLE(3) = 1.0
    IF (I-1.GT.IMAX.AND.Y(I-1).EQ.0.0.AND.Y(I).GT.Y(I-1))
$   HOLE(2) = 1.0
20  CONTINUE
C
C   MORE PLACEMENT OF NOZZLES
C   IF (IFLAG.EQ.1) THEN
      WRITE (6,*) ' '
      WRITE (6,22) NOZZLE
      IF (XNOZ(1).NE.XNOZ(2)) THEN
        WRITE (6,23) PLACE(1),'FRONT ',VERB(1),TYPE(1)
        WRITE (6,23) PLACE(2),'REAR ',VERB(2),TYPE(2)
      ELSE
        WRITE (6,23) PLACE(1),'CENTER',VERB(1),TYPE(1)
      END IF
    END IF
  END IF
C
C   CALCULATE PARAMETERS THAT DO NOT CHANGE AS AREAS ARE BEING
C   CALCULATED (IE. FOUNTAIN CORE POSITION, SLOPES, ANGLES).
C   TWO NOZZLES
    IF (NOZZLE.EQ.2) THEN
      XCORE = (XNOZ(1) + XNOZ(2))/2.0
      YCORE = 0.0
      IF (XNOZ(1).NE.XNOZ(2)) THEN
        E1 = (XNOZ(2) - XNOZ(1))/2.0
        XSPAN = XCORE
      ELSE
        E1 = YNOZ(1)
        IF (YXMIN.EQ.0.0) THEN
          SPAN(3) = XCORE - XMIN
          MSPAN(3) = SPAN(3)
        END IF
        IF (YXMAX.EQ.0.0) THEN
          SPAN(2) = XMAX - XCORE
          MSPAN(2) = SPAN(2)
        END IF
      END IF
      THETA1 = 3.1415926/2.0
    C
    C   THREE AND FOUR NOZZLES
    ELSE
      C   SLOPE OF IMPORTANT LINES
      MLINE2 = (YNOZ(2) - YNOZ(1))/(XNOZ(2) - XNOZ(1))
      IF (MLINE2.NE.0.0) THEN
        MLSPAN = -1.0/MLINE2
        MLINE1 = MLSPAN
        MLINE3 = MLSPAN
      END IF
      C   Y-INTERCEPT OF IMPORTANT LINES
      BLINE1 = -MLINE1 * XNOZ(1) + YNOZ(1)
      BLINE2 = -MLINE2 * XNOZ(1) + YNOZ(1)
      BLINE3 = -MLINE3 * XNOZ(2) + YNOZ(2)
      BLSPAN = -MLSPAN * (XNOZ(1)+XNOZ(2))/2.0 + (YNOZ(1)+YNOZ(2))/2.0
      XCORE = -(BLSPAN / MLSPAN)
      YCORE = 0.0
      M1 = (YCORE - YNOZ(1))/(XCORE - XNOZ(1))
      M2 = (YNOZ(2) - YCORE)/(XNOZ(2) - XCORE)
      THETA4 = ABS(ATAN(M1))

```

```

    THETA2 = ABS(ATAN(M2))
    IF (ATAN(MLSPAN).GT.0.0) THEN
        THETA1 = ATAN(MLSPAN) - THETA2
    ELSE
        THETA1 = 3.1415926 - THETA2 + ATAN(MLSPAN)
    END IF
    THETA3 = 3.1415926 - THETA1 - THETA2 - THETA4
    E1 = DIST(XNOZ(1),YNOZ(1),XNOZ(2),YNOZ(2))/2.0
    E3 = E1
    E4 = YNOZ(1)
    E2 = YNOZ(2)
    IF (NOZZLE.EQ.3.AND.YNOZ(1).GT.YNOZ(2)) THEN
        E2 = YNOZ(1)
        E4 = 0.0
    END IF
    IF (YXMAX.EQ.0.0) THEN
        SPAN(2) = XMAX - XNOZ(2)
        MSPAN(2) = SPAN(2)
    END IF
    IF (YXMIN.EQ.0.0) THEN
        SPAN(3) = XNOZ(1) - XMIN
        MSPAN(3) = SPAN(3)
        IF (YNOZ(2).EQ.0.0) THEN
            SPAN(2) = SPAN(3)
            MSPAN(2) = SPAN(2)
            SPAN(3) = 0.0
            MSPAN(3) = 0.0
        ELSE IF (YNOZ(1).EQ.0.0) THEN
            SPAN(3) = 0.0
            MSPAN(3) = 0.0
        END IF
    END IF
END IF
END IF
END IF
C
C CALCULATE WIDTH OF EACH SLICE.
    DX = XMAX/NSLICE
C SET SLICER AT NOSE AND BEGIN MAKING SLICES ALONG THE THE YSLICE(N) OF
C THE AIRCRAFT.
C   WRITE (6,32)
    N = 0
    X(NPTS+1) = X(NPTS)
    II = 1
30  I = II
    N = N + 1
    IF (N-1 .EQ. 0) THEN
        PSLICE = XSLICE(1)
    ELSE
        PSLICE = XSLICE(N-1)
    END IF
C
C DX IS GOING TO THE RIGHT WITH NO TURNS IN SIGHT
    IF ((X(I).LT.X(I+1)).AND.(X(I+1).LE.X(I+2))) THEN
C     THE NEXT SLICE COMES BEFORE THE NEXT OUTLINE POINT
        IF ((PSLICE+DX).LT.X(I+1)) THEN
            DXX = DX
            II = I
C     THE NEXT SLICE OCCURS AFTER THE NEXT OUTLINE POINT AND BEFORE
C     THE POINT AFTER THAT.

```

```

      ELSE IF (((PSLICE+DX).GE.X(I+1)).AND.
$      ((PSLICE+DX).LE.X(I+2))) THEN
          DXX = DX
          II = I + 1
C      THE NEXT SLICE OCCURS AFTER THE NEXT TWO OUTLINE POINTS
      ELSE IF (((PSLICE+DX).GT.X(I+2)).AND.
$      (X(I+1).LT.X(I+2))) THEN
          II = I + 1
          DXX = ((X(II) + X(II+1))/2.0) - PSLICE
      ELSE IF (X(I+1).EQ.X(I+2)) THEN
          GO TO 100
      END IF
      FACT = 1

C
C DX IS GOING TO THE LEFT WITH NO TURNS IN SIGHT
      ELSE IF ((X(I).GT.X(I+1)).AND.(X(I+1).GE.X(I+2))) THEN
C      THE NEXT SLICE OCCURS BEFORE THE NEXT OUTLINE POINT
      IF ((PSLICE-DX).GT.X(I+1)) THEN
          DXX = -DX
          II = I
C      THE NEXT SLICE OCCURS AFTER THE NEXT OUTLINE POINT AND BEFORE
C      THE POINT AFTER THAT.
      ELSE IF (((PSLICE-DX).LT.X(I+1)).AND.
$      ((PSLICE-DX).GT.X(I+2))) THEN
          DXX = -DX
          II = I + 1
C      THE NEXT SLICE OCCURS AFTER THE NEXT TWO OUTLINE POINTS
      ELSE IF (((PSLICE-DX).LT.X(I+2)).AND.
$      (X(I+1).GT.X(I+2))) THEN
          II = I + 1
          DXX = ((X(II) + X(II+1))/2.0) - PSLICE
      ELSE IF (X(I+1).EQ.X(I+2)) THEN
          GO TO 100
      END IF
      FACT = -1

C
C DX IS GOING TO THE RIGHT AND A TURN IS COMING UP.
      ELSE IF ((X(I).LT.X(I+1)).AND.(X(I+1).GT.X(I+2))) THEN
C      THE NEXT SLICE OCCURS BEFORE THE NEXT OUTLINE POINT
      IF ((PSLICE+DX).LE.X(I+1)) THEN
          DXX = DX
          II = I
          FACT = 1
      ELSE
          IF (PSLICE.LT.X(I+2)) THEN
              II = I + 1
              DXX = ((X(II)+X(II+1))/2.0) - PSLICE
          ELSE IF (PSLICE.GT.((X(I+1)+X(I+2))/2.0)) THEN
              II = I + 1
              DXX = 0.0
          ELSE
              II = I + 1
              DXX = ((X(II)+X(II+1))/2.0) - PSLICE
          END IF
          FACT = -1
      END IF

C
C DX IS GOING TO THE LEFT AND A TURN IS COMING UP.

```

```

ELSE IF ((X(I+1).LT.X(I)).AND.(X(I+2).GT.X(I+1))) THEN
  IF ((PSLICE-DX).GE.X(I+1)) THEN
    DXX = -DX
    II = I
    FACT = -1
  ELSE
    IF (PSLICE.GT.X(I+2)) THEN
      II = I + 1
      DXX = ((X(II)+X(II+1))/2.0) - PSLICE
    ELSE IF (PSLICE.LT.((X(I+1)+X(I+2))/2.0)) THEN
      II = I + 1
      DXX = 0.0
    ELSE
      II = I + 1
      DXX = ((X(II)+X(II+1))/2.0) - PSLICE
    END IF
    FACT = 1
  END IF
END IF

C
C CALCULATE NEW SLICE POSITION GIVEN 'DXX' FROM ABOVE AND DETERMINE THE
C CORRESPONDING YSLICE(N) FOR THAT POSITION.
  IF (N-1 .EQ. 0) THEN
    XSLICE(N) = DXX
  ELSE
    XSLICE(N) = XSLICE(N-1) + DXX
  END IF
  CALL LINTERP(XSLICE(N),X(II),X(II+1),Y(II),Y(II+1),yslice(n))
  IF (N.GE.2000) THEN
    SDAERR = 6
    IF (IFLAG.EQ.1) WRITE (6,*) 'THE NUMBER OF INTERNAL SLICES HAS
$ EXCEEDED THE MAXIMUM NUMBER OF SLICES (2000). PLEASE REDUCE THE
$ NUMBER OF SLICES.'
    GO TO 500
  END IF

C
C DETERMINE WHERE THE SLICER IS ON THE PLANFORM AND SET THE VALUES OF
C A,B, AND C TO THEIR RESPECTIVE VALUES. ALSO DETERMINE THE POSITIONS
C OF THE SPANWISE AND MAXIMUM EXTENT OF THE PLANFORM ON THE FOUNTAIN
C ARM.
C TWO NOZZLES
C IF (NOZZLE.EQ.2) THEN
C
C ONE NOZZLE IN THE FRONT AND ONE IN THE REAR.
C IF (YNOZ(1).EQ.0.0) THEN
  IF ((XSLICE(N).GT.XNOZ(1)).AND.(XSLICE(N).LT.XNOZ(2)))
$ THEN
  A = YSLICE(N)
  B = 0.0
  ELSE
  A = 0.0
  B = 0.0
  END IF
  DETERMINE MAXIMUM SPANWISE AND SPANWISE EXTENT OF PLANFORM
  ON FOUNTAIN ARM CENTERLINE.
  IF ((XSLICE(N).GE.XSPAN).AND.(XSLICE(N-1).LT.XSPAN))
$ THEN
  CALL LINTERP(XSPAN,XSLICE(N),XSLICE(N-1),YSLICE(N),

```

```

$      YSLICE(N-1),ytemp)
      IF (SPAN(1).LT.YTEMP) SPAN(1) = YTEMP
      END IF
      IF (MSPAN(1).LT.A) MSPAN(1) = A
      ATYPE = 1

C
C      TWO NOZZLES SIDE BY SIDE
      ELSE
        IF (YSLICE(N).LT.YNOZ(1)) THEN
          A = YSLICE(N)
          B = 0.0
        ELSE
          A = YNOZ(1)
          B = 0.0
        END IF
        DETERMINE MAXIMUM SPANWISE AND SPANWISE EXTENT OF PLANFORM
        ON FOUNTAIN ARM CENTERLINE.
        IF (YSLICE(N).LT.E1.AND.YXMIN.NE.0.0) THEN
          IF (N.EQ.1) THEN
            SPAN(1) = XCORE - X(1)
            SPAN(2) = X(NPTS) - XCORE
          ELSE IF (YSLICE(N).EQ.0.0) THEN
            IF (XCORE-XSLICE(N).GT.0.0) THEN
              IF (SPAN(1).LT.(XCORE-XSLICE(N)))
                SPAN(1) = XCORE - XSLICE(N)
              ELSE IF (XSLICE(N)-XCORE.GT.0.0) THEN
                IF (SPAN(2).LT.XSLICE(N)-XCORE)
                  SPAN(2) = XSLICE(N) - XCORE
              END IF
            END IF
            IF ((XSLICE(N)-XCORE).LT.(-MSPAN(1))) THEN
              MSPAN(1) = XCORE - XSLICE(N)
            ELSE IF ((XSLICE(N)-XCORE).GT.(MSPAN(2))) THEN
              MSPAN(2) = XSLICE(N) - XCORE
            END IF
          END IF
          If slicer is before the nozzles then add to STOT(3)
          else add to STOT(2)
          IF (XSLICE(N).LT.XNOZ(1)) THEN
            ATYPE = 3
          ELSE
            ATYPE = 2
          END IF
          END IF
          C = 0.0

C
C      THREE NOZZLE
      ELSE IF (NOZZLE.EQ.3) THEN
        LINE2 = MLINE2*XSLICE(N)+BLINE2
        LINE1 = MLINE1*XSLICE(N)+BLINE1
        LINE3 = MLINE3*XSLICE(N)+BLINE3
        LSPAN = LSPAN
        LSPAN = MLSPAN*XSLICE(N)+BLSPAN
        IF (YNOZ(1).GT.0.0) THEN
          LINE4 = YNOZ(1)
        ELSE
          LINE4 = YNOZ(2)
        END IF
      END IF

```

C

ONE NOZZLE IN THE FRONT AND TWO NOZZLE IN THE REAR
 IF (LINE1.LT.LINE3) THEN

IF (LINE1.GE.LINE2) THEN

IF (LINE1.LT.YSLICE(N)) THEN

IF (LINE3.LT.YSLICE(N)) THEN

A = LINE3

B = LINE1

ELSE

A = YSLICE(N)

B = LINE1

END IF

ELSE

A = 0.0

B = 0.0

END IF

C = 0.0

ATYPE = 1

ELSE IF (LINE3.GE.LINE2) THEN

IF (LINE3.LT.YSLICE(N)) THEN

A = LINE3

B = LINE2

ELSE IF (LINE2.LT.YSLICE(N)) THEN

A = YSLICE(N)

B = LINE2

ELSE

A = 0.0

B = 0.0

END IF

C = 0.0

ATYPE = 1

ELSE IF (LINE4.LT.LINE2) THEN

IF (LINE4.GT.YSLICE(N)) THEN

A = YSLICE(N)

ELSE

A = LINE4

END IF

B = 0.0

C = 0.0

ATYPE = 2

IF (YSLICE(N).EQ.0.0.AND.SPAN(2).LT.XSLICE(N)-XNOZ(2))

\$ SPAN(2) = XSLICE(N)-XNOZ(2)

\$ IF (((YSLICE(N).LT.LINE4).AND.((XSLICE(N)-XNOZ(2)).GT.

\$ MSPAN(2)))) MSPAN(2) = XSLICE(N) - XNOZ(2)

END IF

C

C

Skip test for X,YSPAN if N=1, because subscript out of range

IF (N.NE.1) THEN

IF ((LSPANO-YSLICE(N-1))*(LSPAN-YSLICE(N)).LE.0.0) THEN

CALL CROSSIN(XSLICE(N-1),YSLICE(N-1),XSLICE(N),

\$ YSLICE(N),XSLICE(N-1),LSPANO,XSLICE(N),LSPAN,

\$ xtemp,ytemp)

IF (YSPAN.LT.YTEMP) THEN

YSPAN = YTEMP

XSPAN = XTEMP

END IF

END IF

C

END IF

```

C      TWO NOZZLE IN THE FRONT AND ONE IN THE REAR.
      ELSE
        IF (LINE3.GE.LINE2) THEN
          IF (LINE3.LT.YSLICE(N)) THEN
            IF (LINE1.LT.YSLICE(N)) THEN
              A = LINE1
              B = LINE3
            ELSE
              A = YSLICE(N)
              B = LINE3
            END IF
          ELSE
            A = 0.0
            B = 0.0
          END IF
          C = 0.0
          ATYPE = 1
        ELSE IF (LINE1.GE.LINE2) THEN
          IF (LINE1.LT.YSLICE(N)) THEN
            A = LINE1
            B = LINE2
          ELSE IF (LINE2.LT.YSLICE(N)) THEN
            A = YSLICE(N)
            B = LINE2
          ELSE
            A = 0.0
            B = 0.0
          END IF
          C = 0.0
          ATYPE = 1
        ELSE IF (LINE4.LT.LINE2) THEN
          IF (LINE4.GT.YSLICE(N)) THEN
            A = YSLICE(N)
          ELSE
            A = LINE4
          END IF
          B = 0.0
          C = 0.0
          ATYPE = 2
          IF (YSLICE(N).EQ.0.0.AND.SPAN(2).LT.XNOZ(1)-XSLICE(N))
$      SPAN(2) = XNOZ(1) - XSLICE(N)
          IF (((YSLICE(N).LT.LINE4).AND.((XSLICE(N)-XNOZ(1)).LT.
$      (-MSPAN(2)))) MSPAN(2) = XNOZ(1) - XSLICE(N)
          END IF
        END IF
      END IF
C
C      FOUR NOZZLES
      ELSE
        LINE2 = MLINE2*XSLICE(N)+BLINE2
        IF (YNOZ(1).NE.YNOZ(2)) THEN
          LINE1 = MLINE1*XSLICE(N)+BLINE1
          LINE3 = MLINE3*XSLICE(N)+BLINE3
          LSPANO = LSPAN
          LSPAN = MLSPAN*XSLICE(N)+BLSPAN
        END IF
        LINE4 = YNOZ(2)
        LINE5 = YNOZ(1)
C      BOTH NOZZLES ARE THE SAME DISTANCE FROM THE CENTER LINE.

```

```

IF (LINE5.EQ.LINE4) THEN
  IF (XSLICE(N).LT.XNOZ(1)) THEN
    IF (LINE5.LT.YSLICE(N)) THEN
      A = LINE5
    ELSE
      A = YSLICE(N)
    END IF
    B = 0.0
    C = 0.0
    ATYPE = 3
  ELSE IF (XSLICE(N).GT.XNOZ(2)) THEN
    IF (LINE4.LT.YSLICE(N)) THEN
      A = LINE4
    ELSE
      A = YSLICE(N)
    END IF
    B = 0.0
    C = 0.0
    ATYPE = 2
  ELSE IF (LINE2.LT.YSLICE(N)) THEN
    A = YSLICE(N)
    B = LINE2
    C = 0.0
    ATYPE = 1
  END IF
C  THE REAR NOZZLE ARE CLOSER TOGETHER THAN THE FRONT NOZZLE.
ELSE IF (LINE5.GT.LINE4) THEN
  IF (LINE2.GT.LINE5) THEN
    IF (LINE5.LT.YSLICE(N)) THEN
      A = LINE5
    ELSE
      A = YSLICE(N)
    END IF
    B = 0.0
    C = 0.0
    ATYPE = 3
  ELSE IF ((LINE2.LE.LINE5).AND.(LINE1.LT.YSLICE(N))) THEN
    A = LINE1
    B = LINE2
    C = 0.0
    ATYPE = 1
  ELSE IF (LINE2.LT.LINE4) THEN
    IF (LINE3.LT.YSLICE(N)) THEN
      A = YSLICE(N)
      B = LINE3
      C = LINE4
      ATYPE = 4
    ELSE IF (LINE4.LT.YSLICE(N)) THEN
      A = LINE4
      B = 0.0
      C = 0.0
      ATYPE = 2
    ELSE
      A = YSLICE(N)
      B = 0.0
      C = 0.0
      ATYPE = 2
    END IF
  END IF

```



```

ELSE IF (LINE2.LT.YSLICE(N)) THEN
  A = YSLICE(N)
  B = LINE2
  C = 0.0
  ATYPE = 1
ELSE
  A = 0.0
  B = 0.0
  C = 0.0
END IF
C THE FRONT NOZZLE ARE CLOSER TOGETHER THAN THE REAR NOZZLE.
ELSE
  IF (LINE2.GT.LINE4) THEN
    IF (LINE4.LT.YSLICE(N)) THEN
      A = LINE4
    ELSE
      A = YSLICE(N)
    END IF
    B = 0.0
    C = 0.0
    ATYPE = 2
  ELSE IF ((LINE2.LE.LINE4).AND.(LINE3.LT.YSLICE(N))) THEN
    A = LINE3
    B = LINE2
    C = 0.0
    ATYPE = 1
  ELSE IF (LINE2.LT.LINE5) THEN
    IF (LINE1.LT.YSLICE(N)) THEN
      A = YSLICE(N)
      B = LINE1
      C = LINE5
      ATYPE = 5
    ELSE IF (LINE5.LT.YSLICE(N)) THEN
      A = LINE5
      B = 0.0
      C = 0.0
      ATYPE = 3
    ELSE
      A = YSLICE(N)
      B = 0.0
      C = 0.0
      ATYPE = 3
    END IF
  ELSE IF (LINE2.LT.YSLICE(N)) THEN
    A = YSLICE(N)
    B = LINE2
    C = 0.0
    ATYPE = 1
  ELSE
    A = 0.0
    B = 0.0
    C = 0.0
  END IF
END IF
IF (YNOZ(1).EQ.YNOZ(2)) THEN
  XSPAN = XCORE
  IF ((XSLICE(N).GE.XSPAN).AND.(XSLICE(N-1).LT.XSPAN))
S THEN

```

```

        CALL LINTERP(XSPAN,XSLICE(N),XSLICE(N-1),YSLICE(N),
$         YSLICE(N-1),ytemp)
        IF (SPAN(1).LT.YTEMP) SPAN(1) = YTEMP
        END IF
        IF (MSPAN(1).LT.A) MSPAN(1) = A
    ELSE
        IF (YSLICE(N).EQ.0.0) THEN
            IF (SPAN(2).LT.XSLICE(N)-XNOZ(2)) SPAN(2) = XSLICE(N) -
$             XNOZ(2)
            IF (SPAN(3).LT.XNOZ(1)-XSLICE(N)) SPAN(3) = XNOZ(1) -
$             XSLICE(N)
        END IF
        IF (((YSLICE(N).LT.LINE4).AND.((XSLICE(N)-XNOZ(2)).GT.
$         MSPAN(2)))) MSPAN(2) = XSLICE(N) - XNOZ(2)
        IF (((YSLICE(N).LT.LINE5).AND.((XNOZ(1)-XSLICE(N)).GT.
$         MSPAN(3)))) MSPAN(3) = XNOZ(1) - XSLICE(N)
        END IF
    END IF
C
C CALCULATE THE X & Y COORDINATES THAT CORRESPOND TO THE MAXIMUM
C SPANWISE EXTENT OF PLANFORM ON FOUNTAIN ARM CENTERLINE IN REGION 1
C (THE SIDE OF THE AIRCRAFT.)
    IF (NOZZLE.GT.2.AND.A.EQ.YSLICE(N).AND.B.NE.0.0) THEN
        IF ((LSPANO-YSLICE(N-1))*(LSPAN-YSLICE(N)).LE.0.0) THEN
            CALL CROSSIN(XSLICE(N-1),YSLICE(N-1),XSLICE(N),YSLICE(N),
$             XSLICE(N-1),LSPANO,XSLICE(N),LSPAN,xtemp,ytemp)
            IF (YSPAN.LT.YTEMP) THEN
                YSPAN = YTEMP
                XSPAN = XTEMP
            END IF
        END IF
        BTEST = YSLICE(N) - MLINE2 * XSLICE(N)
        CALL LINECRO(MLINE1,BLINE1,MLINE2,BTEST,xtemp,ytemp)
        IF (YTEMP.GT.YTEMPOLD) THEN
            YTEMPOLD = YTEMP
            XMSPAN = XSLICE(N)
            YMSPAN = YSLICE(N)
        END IF
    END IF
C
C CALCULATE THE AREA FOR EACH DXX AND ADD OR SUBTRACT THAT FROM THE
C TOTAL AREA IN EACH REGION.
    IF (DXX.EQ.0.0) DXX = DX
    DXX = ABS(DXX)
    CALL INTEGRA(A,B,C,DXX,SAB,SC)
    IF (ATYPE.EQ.4) THEN
        STOT(1) = STOT(1) + FACT * SAB
        STOT(2) = STOT(2) + FACT * SC
    ELSE IF (ATYPE.EQ.5) THEN
        STOT(1) = STOT(1) + FACT * SAB
        STOT(3) = STOT(3) + FACT * SC
    ELSE
        STOT(ATYPE) = STOT(ATYPE) + FACT * SAB
    END IF
C
    GO TO 30
C
C CALCULATE MAXIMUM SPANWISE AND SPANWISE EXTENT OF PLANFORM ON FOUNTAIN

```

```

C AREA CENTERLINE FOR REGION 1 (THE SIDE OF THE AIRCRAFT.)
100 IF (NOZZLE.GT.2 .AND. YMSPAN.NE.0.0) THEN
    CALL PERPDIS(XSPAN, YSPAN, MLINE2, BLINE2, span(1))
    CALL PERPDIS(XMSPAN, YMSPAN, MLINE2, BLINE2, mspan(1))
END IF

C
C CALCULATE AND SUBTRACT THE AREAS OF THE NOZZLE FROM THE TOTAL
C AREAS
IF (IFLAG .EQ. 1) THEN
    NOZARE(1) = RAD(1)**2.0 * 3.1415926
    NOZARE(2) = RAD(2)**2.0 * 3.1415926
ELSE
    NOZARE(1) = S_F/NUM_F
    NOZARE(2) = S_R/NUM_R
END IF
IF (TYPE(1).EQ.'INTERNAL.') THEN
    IF (XNOZ(1).EQ.XNOZ(2)) THEN
        STOT(2) = STOT(2) - NOZARE(1)/4.0
        STOT(3) = STOT(3) - NOZARE(1)/4.0
    ELSE
        STOT(1) = STOT(1) - NOZARE(1)/4.0
    END IF
    IF (NOZZLE.EQ.3) THEN
        IF (YNOZ(2).EQ.0.0) STOT(2) = STOT(2) - NOZARE(1)/4.0
    ELSE IF (NOZZLE.EQ.4) THEN
        STOT(3) = STOT(3) - NOZARE(1)/4.0
    END IF
END IF
IF (TYPE(2).EQ.'INTERNAL.') THEN
    IF (XNOZ(1) .NE. XNOZ(2)) STOT(1) = STOT(1) - NOZARE(2)/4.0
    IF (NOZZLE.EQ.3) THEN
        IF (YNOZ(1).EQ.0.0) STOT(2) = STOT(2) - NOZARE(2)/4.0
    ELSE IF (NOZZLE.EQ.4) THEN
        STOT(2) = STOT(2) - NOZARE(2)/4.0
    END IF
END IF

C
C FINISH CALCULATIONS ON ALL AREAS TO BE SENT BACK TO CALLING
C ROUTINE.
C
IF (NOZZLE.EQ.2) THEN
    IF (XNOZ(1) .NE. XNOZ(2)) THEN
        SP1 = STOT(1)
        SPP1 = 2 * E1 * MSPAN(1)
    ELSE
        SP2 = (1.0 - HOLE(2)) * 2.0 * STOT(2)
        SP3 = (1.0 - HOLE(3)) * 2.0 * STOT(3)
        SPP2 = (1.0 - HOLE(2)) * 2.0 * E1 * MSPAN(2)
        SPP3 = (1.0 - HOLE(3)) * 2.0 * E1 * MSPAN(3)
    END IF
ELSE
    IF (NOZZLE.EQ.3.AND.YNOZ(2).EQ.0.0) THEN
        HOLE(1) = HOLE(2)
        HOLE(2) = HOLE(3)
        HOLE(3) = HOLE(1)
    END IF
    SP1 = STOT(1)
    SP2 = (1.0 - HOLE(2)) * 2.0 * STOT(2)
    SP3 = (1.0 - HOLE(3)) * 2.0 * STOT(3)

```

```

SPP2 = (1.0 - HOLE(2)) * 2.0 * E2 * MSPAN(2)
SPP3 = (1.0 - HOLE(3)) * 2.0 * E4 * MSPAN(3)

C
C
CALCULATE SPP1
IF (YMSPAN .EQ. 0.0) THEN
  SPP1 = 0.0
ELSE
  X1 = XNOZ(1)
  Y1 = YNOZ(1)
  X2 = XNOZ(2)
  Y2 = YNOZ(2)
  MMSPAN = -1.0/MLINE1
  BMSPAN = YMSPAN - MMSPAN * XMSPAN
  CALL LINECRO(MMSPAN,BMSPAN,MLINE1,BLINE1,x4,y4)
  CALL LINECRO(MMSPAN,BMSPAN,MLINE3,BLINE3,x3,y3)
  SPP1 = .5 * (X1*Y2 + X2*Y3 + X3*Y4 + X4*Y1 - Y1*X2 - Y2*X3 -
$      Y3*X4 - Y4*X1)
END IF
END IF

C
C
MAKE FINAL CALCULATIONS FOR ALL SPANS AND MSPANS.
IF (SPAN(1) .LT. 0.0) SPAN(1) = 0.0
SPAN(2) = (1.0 - HOLE(2)) * SPAN(2)
SPAN(3) = (1.0 - HOLE(3)) * SPAN(3)
MSPAN(2) = (1.0 - HOLE(2)) * MSPAN(2)
MSPAN(3) = (1.0 - HOLE(3)) * MSPAN(3)
-----
C Assign the correct variables to the variables from the calling
C routine COPPIE
C
C Calculate estimated nose configuration
C
C SCALE3 is a percentage measured from the nozzle to the tip of the
C nose.
C
IF (SCALE3 .NE. 1.0 .OR. SCALE .EQ. 9999.) THEN
$  SCALE = 1. - (-.00129228 + .31836 * (1. - SCALE3) + 4.41157 *
$  (1. - SCALE3)**2. - 10.376 * (1. - SCALE3)**3. +
$  6.65561 * (1. - SCALE3)**4.)
ELSE
  SCALE = 1.0
END IF

C
C Four nozzle configuration
C
C
C      222      X      X      222
C      2      2      X      X      2      2
C      2      2      X      X      2
C      2      2      X      X      2
C      2      2      X      X      2
C      2      2      X      X      2
C      2      2      X      X      2
C      22222      X      X      22222
C
C
IF (NUM_F .EQ. 2 .AND. NUM_R .EQ. 2) THEN
C Half distance between fountain arms
IF (E_1 .EQ. 9999.) E_1 = E1
IF (E_3 .EQ. 9999.) E_3 = E4
IF (E_4 .EQ. 9999.) E_4 = E2

```

```

C      Half angles between fountain arms
      IF (THA_1 .EQ. 9999.) THA_1 = THETA1
      IF (THA_3 .EQ. 9999.) THA_3 = THETA4
      IF (THA_4 .EQ. 9999.) THA_4 = THETA2
C      Half length of fountain arms
      IF (Y_1 .EQ. 9999.) Y_1 = SPAN(1)
      IF (Y_3 .EQ. 9999.) Y_3 = SPAN(3) * SCALE3
      IF (Y_4 .EQ. 9999.) Y_4 = SPAN(2)
C      Maximum spanwise extent of planform
      IF (YP_1 .EQ. 9999.) YP_1 = MSPAN(1)
      IF (YP_3 .EQ. 9999.) YP_3 = MSPAN(3) * SCALE3
      IF (YP_4 .EQ. 9999.) YP_4 = MSPAN(2)
C      Area affected by fountain arm
      IF (S_FA1 .EQ. 9999.) S_FA1 = SP1
      IF (S_FA3 .EQ. 9999.) S_FA3 = SP3 * SCALE
      IF (S_FA4 .EQ. 9999.) S_FA4 = SP2
C      Potential area affected by fountain arm
      IF (SP_FA1 .EQ. 9999.) SP_FA1 = SPP1
      IF (SP_FA3 .EQ. 9999.) SP_FA3 = SPP3 * SCALE3
      IF (SP_FA4 .EQ. 9999.) SP_FA4 = SPP2

C      Three nozzle configuration (One nozzle in front, two in rear)
      1      X      X      222
      11     X      X      2  2
      1 1    X X      2
      1      X      2
      1      X X      2
      1      X      X      2
      11111  X      X      22222

C      ELSE IF (NUM_F .EQ. 1 .AND. NUM_R .EQ. 2) THEN
C
C      Pieces at _4 have been moved to _3 due to hov_ge which when
C      making calculations for a three nozzle configuration uses _3
C      instead of _4 in its calculations no matter which configuration
C      is being used (1 X 2 or 2 X 1).
C
C      Half distance between fountain arms
      IF (E_1 .EQ. 9999.) E_1 = E1
      IF (E_3 .EQ. 9999.) E_3 = E2
      IF (E_4 .EQ. 9999.) E_4 = 0.0
C      Half angles between fountain arms
      IF (THA_1 .EQ. 9999.) THA_1 = THETA1
      IF (THA_3 .EQ. 9999.) THA_3 = THETA2
      IF (THA_4 .EQ. 9999.) THA_4 = 0.0
C      Half length of fountain arms
      IF (Y_1 .EQ. 9999.) Y_1 = SPAN(1)
      IF (Y_3 .EQ. 9999.) Y_3 = SPAN(2)
      IF (Y_4 .EQ. 9999.) Y_4 = 0.0
C      Maximum spanwise extent of planform
      IF (YP_1 .EQ. 9999.) YP_1 = MSPAN(1)
      IF (YP_3 .EQ. 9999.) YP_3 = MSPAN(2)
      IF (YP_4 .EQ. 9999.) YP_4 = 0.0
C      Area affected by fountain arm
      IF (S_FA1 .EQ. 9999.) S_FA1 = SP1
      IF (S_FA3 .EQ. 9999.) S_FA3 = SP2
      IF (S_FA4 .EQ. 9999.) S_FA4 = 0.0

```

```

C      Potential area affected by fountain arm
      IF (SP_FA1 .EQ. 9999.) SP_FA1 = SPP1
      IF (SP_FA3 .EQ. 9999.) SP_FA3 = SPP2
      IF (SP_FA4 .EQ. 9999.) SP_FA4 = 0.0

```

```

C      Three nozzle configuration (Two nozzles in front, one in rear)
C
C

```

```

C      222      X      X      1
C      2      2      X      X      11
C      2      X      X      1 1
C      2      X      1
C      2      X      X      1
C      2      X      X      1
C      22222      X      X      11111
C

```

```

C      ELSE IF (NUM_F .EQ. 2 .AND. NUM_R .EQ. 1) THEN
C

```

```

C      Half distance between fountain arms
      IF (E_1 .EQ. 9999.) E_1 = E1
      IF (E_3 .EQ. 9999.) E_3 = E2
      IF (E_4 .EQ. 9999.) E_4 = 0.0
C      Half angles between fountain arms
      IF (THA_1 .EQ. 9999.) THA_1 = THETA1
      IF (THA_3 .EQ. 9999.) THA_3 = THETA4
      IF (THA_4 .EQ. 9999.) THA_4 = THETA2
C      Half length of fountain arms
      IF (Y_1 .EQ. 9999.) Y_1 = SPAN(1)
      IF (Y_3 .EQ. 9999.) Y_3 = SPAN(2) * SCALE3
      IF (Y_4 .EQ. 9999.) Y_4 = 0.0
C      Maximum spanwise extent of planform
      IF (YP_1 .EQ. 9999.) YP_1 = MSPAN(1)
      IF (YP_3 .EQ. 9999.) YP_3 = MSPAN(2) * SCALE3
      IF (YP_4 .EQ. 9999.) YP_4 = 0.0
C      Area affected by fountain arm
      IF (S_FA1 .EQ. 9999.) S_FA1 = SP1
      IF (S_FA3 .EQ. 9999.) S_FA3 = SP2 * SCALE
      IF (S_FA4 .EQ. 9999.) S_FA4 = 0.0
C      Potential area affected by fountain arm
      IF (SP_FA1 .EQ. 9999.) SP_FA1 = SPP1
      IF (SP_FA3 .EQ. 9999.) SP_FA3 = SPP2 * SCALE3
      IF (SP_FA4 .EQ. 9999.) SP_FA4 = 0.0

```

```

C      Two nozzle configuration (One nozzle in front, one in rear)
C
C

```

```

C      1      X      X      1
C      11      X      X      11
C      1 1      X      X      1
C      1      X      1
C      1      X      X      1
C      1      X      X      1
C      11111      X      X      11111
C

```

```

C      ELSE IF (NUM_F .EQ. 1 .AND. NUM_R .EQ. 1) THEN
C

```

```

C      Half distance between fountain arms
      IF (E_1 .EQ. 9999.) E_1 = E1
      IF (E_3 .EQ. 9999.) E_3 = 0.0
      IF (E_4 .EQ. 9999.) E_4 = 0.0

```

```

C      Half angles between fountain arms
      IF (THA_1 .EQ. 9999.) THA_1 = THETA1
      IF (THA_3 .EQ. 9999.) THA_3 = 0.0
      IF (THA_4 .EQ. 9999.) THA_4 = 0.0
C      Half length of fountain arms
      IF (Y_1 .EQ. 9999.) Y_1 = SPAN(1)
      IF (Y_3 .EQ. 9999.) Y_3 = 0.0
      IF (Y_4 .EQ. 9999.) Y_4 = 0.0
C      Maximum spanwise extent of planform
      IF (YP_1 .EQ. 9999.) YP_1 = MSPAN(1)
      IF (YP_3 .EQ. 9999.) YP_3 = 0.0
      IF (YP_4 .EQ. 9999.) YP_4 = 0.0
C      Area affected by fountain arm
      IF (S_FA1 .EQ. 9999.) S_FA1 = 2.0 * SP1
      IF (S_FA3 .EQ. 9999.) S_FA3 = 0.0
      IF (S_FA4 .EQ. 9999.) S_FA4 = 0.0
C      Potential area affected by fountain arm
      IF (SP_FA1 .EQ. 9999.) SP_FA1 = 2.0 * SPP1
      IF (SP_FA3 .EQ. 9999.) SP_FA3 = 0.0
      IF (SP_FA4 .EQ. 9999.) SP_FA4 = 0.0

```

```

C      Two nozzle configuration (Two nozzles in front)

```

```

C      222      X      X      000
C      2      2      X      X      0 00
C      2      X X      0 0
C      2      X      0 0 0
C      2      X X      0 0
C      2      X      X      00 0
C      22222      X      X      000

```

```

C      ELSE IF (NUM_F .EQ. 2 .AND. NUM_R .EQ. 0) THEN

```

```

C      Half distance between fountain arms
      IF (E_1 .EQ. 9999.) E_1 = E1
      IF (E_3 .EQ. 9999.) E_3 = 0.0
      IF (E_4 .EQ. 9999.) E_4 = 0.0
C      Half angles between fountain arms
      IF (THA_1 .EQ. 9999.) THA_1 = THETA1
      IF (THA_3 .EQ. 9999.) THA_3 = 0.0
      IF (THA_4 .EQ. 9999.) THA_4 = 0.0
C      Half length of fountain arms
      IF (Y_1 .EQ. 9999.) Y_1 = (SPAN(2) + SPAN(3) * SCALE3) / 2.0
      IF (Y_3 .EQ. 9999.) Y_3 = 0.0
      IF (Y_4 .EQ. 9999.) Y_4 = 0.0
C      Maximum spanwise extent of planform
      IF (YP_1 .EQ. 9999.) YP_1 = (MSPAN(2) + MSPAN(3) * SCALE3)
$      / 2.0
      IF (YP_3 .EQ. 9999.) YP_3 = 0.0
      IF (YP_4 .EQ. 9999.) YP_4 = 0.0
C      Area affected by fountain arm
      IF (S_FA1 .EQ. 9999.) S_FA1 = (SP2 + SP3 * SCALE) / 2.0
      IF (S_FA3 .EQ. 9999.) S_FA3 = 0.0
      IF (S_FA4 .EQ. 9999.) S_FA4 = 0.0
C      Potential area affected by fountain arm
      IF (SP_FA1 .EQ. 9999.) SP_FA1 = (SPP2 + SPP3 * SCALE3) / 2.0
      IF (SP_FA3 .EQ. 9999.) SP_FA3 = 0.0
      IF (SP_FA4 .EQ. 9999.) SP_FA4 = 0.0

```

```

      END IF
C
C   Core coordinates
      X_CORE = XCORE
      Y_CORE = YCORE
C-----
C   Print out everything when necessary
      IF (IFLAG.EQ.1) THEN
C       WRITE (6,35) STOT(1), STOT(2), STOT(3)
       WRITE (6,40) XCORE, YCORE, SP1, SP2, SP3, SPP1, SPP2, SPP3, SPAN(1),
$   SPAN(2), SPAN(3), MSPAN(1), MSPAN(2), MSPAN(3)
C       WRITE (6,*) '          XCORE          YCORE'
C       WRITE (6,*) XCORE, YCORE
C       WRITE (6,*) '          M1  ', '          M2  ', '          M3  '
C       WRITE (6,*) M1, M2, M3
C       WRITE (6,*) '          THETA1', '          THETA2', '          THETA3',
C       $ '          THETA4'
C       WRITE (6,*) THETA1, THETA2, THETA3, THETA4
C       WRITE (6,*) '          E1  ', '          E2  ', '          E3  ',
C       $ '          E4  '
C       WRITE (6,*) E1, E2, E3, E4
22    FORMAT (1X, 'THERE ARE A TOTAL OF ', I1, ' NOZZLE ON THIS
$   AIRCRAFT.', /)
23    FORMAT (4X, 'THERE ', A15, ' IN THE ', A6, ' WHICH ', A3, ' ', A9)
32    FORMAT (1X, 'POINT', 1X, '    XSLICE', 1X, 'YSLICE', 1X, '    DX', 1X,
$   '    S1', 1X, '    S2', 1X, '    S3', 1X, '    A', 1X, '    B')
33    FORMAT (1X, I4, I4, 1X, F6.2, 1X, F6.2, 1X, F6.2, 1X, F6.2, 1X, F6.2, 1X, F6.2,
$   1X, F6.2, 1X, F6.2, 1X, I3, 1X, F3.0)
35    FORMAT (1X, 'STOT(1) = ', F6.2, 2X, 'STOT(2) = ', F6.2, 2X, 'STOT(3) = ',
$   F6.2)
40    FORMAT (/1X, 'XCORE = ', F10.2, 4X, 'YCORE = ', F5.2//1X, 'SP1 = ',
$   F6.2, 5X, 'SP2 = ', F6.2, 5X, 'SP3 = ', F6.2//1X, 'SPP1 = ', F6.2, 5X,
$   'SPP2 = ', F6.2, 5X, 'SPP3 = ', F6.2//1X, 'SPAN1 = ', F6.2, 5X,
$   'SPAN2 = ', F6.2, 5X, 'SPAN3 = ', F6.2//1X, 'MSPAN1 = ', F6.2, 5X,
$   'MSPAN2 = ', F6.2, 5X, 'MSPAN3 = ', F6.2)
      END IF
C
500  RETURN
      END

```

Integra

```

      SUBROUTINE INTEGRA(A,B,C,DX,AREA_AB,AREA_C)
C   THIS SUBROUTINE CALCULATES THE AREA THAT IS BOUNDED BY TWO LINES
C   PLUS ANOTHER ONE AND ZERO.
C       AREA_AB = (A - B) * DX
C       AREA_C = (C - 0) * DX
C
      REAL A, B, C, DX, AREA_AB, AREA_C
      AREA_AB = (A - B) * DX
      AREA_C = C * DX
      RETURN
      END

```

Diabar

```

      Subroutine DIABAR (Iflag, Ierror, X, Y, NPTS, xctr, nangle, dbar)
C-----

```


C ACRONYM: DIABAR since DBAR is some reserved function.

C
C PURPOSE: DBAR is the Angular Mean Diameter of a Planform. This
C routine will calculate DBAR and areas forward and aft of the
C DBAR point. An input file containing planform coordinates
C (in a clockwise direction starting from the nose) is
C required along with the x-position from which to calculate
C DBAR.

C PARAMETERS:

NAME	TYPE	I/O	UNITS	DESCRIPTION
IFLAG	I	I	----	User interaction flag: 1 = user interaction (can run as a stand alone routine. otherwise = no user interaction (all inputs from the calling routine).
IERROR	I	O	----	Error flag returned by DIABAR. If IFLAG = 1, then an explanation of the problem is given at the time of the error. 0 = no errors. 1 = values of dbar & area probably bad. 2 = moved x-position of dbar point/nozzle point to geometric center. 3 = input NANGLE was inappropriate (set to absolute value or 1000). 4 = Points entered in wrong direction, routine aborted. Must enter points from left to right. 5 = 1st and last points not on x-axis, routine aborted. 6 = the sweeping ray intersected more than 4 points or zero points; values returned may be bad.
X (500)	R		feet	X-values of the planform (limit = 500). Values must start from the left and procede counterclockwise for one half of the aircraft (the aircraft is assumed symmetrical).
Y (500)	R		feet	Y-values of the planform (limit = 500).
NPTS	I		----	The number of points read in from the planform data file.
XCTR	R	I	feet	The x-coordinate of the point at which DBAR and/or areas are to be calculated.
NANGLE	I	I	----	Number of angle divisions to use in the calculation of DBAR and/or areas.
DBAR	R	O	feet	Angular Mean Diameter of the planform.
FRAREA	R	O	ft2	The planform area in front (to the left) of the XCTR point.

NAME	TYPE	UNITS	DESCRIPTION
BKAREA	R	O ft2	The planform area behind (to the right) of the XCTR point.
LOCAL VARIABLES (in addition to the above parameters):			
ALPHA	R	radians	Current sweep angle.
AREA1	R	ft2	Temperary variable for the total area.
ARM1	R	ft2	Partial area for the intersection point in consideration.
ARM2	R	ft2	See ARM1.
ARM3	R	ft2	See ARM1.
ARM4	R	ft2	See ARM1.
DELTA	R	radians	The step angle (PI/NANGLE) for the DBAR and area calculations.
GEOCTR	R	feet	The geometric center (average x-value) based on the first and last points of the planform data file.
I	I	----	Temperary counter variable.
IND	I	----	XCTR position flag: 0 = XCTR inside planform. 1 = XCTR behind planform and will cause frarea to equal tlarea and bkarea to equal 0 for output. 2 = XCTR forward of planform and will cause bkarea to equal tlarea and frarea to equal 0 for output.
IND2	I	----	Flag for setting XCTR = GEOCTR: 0 = Leave XCTR as is. 1 = Set XCTR = GEOCTR
IND3	I	----	Trouble message flag (for IFLAG = 1): 0 = No trouble in NINMAR or NINCEP. 1 = NINMAR = 0, and/or NINCEP > 4 or < 1 (IERROR = 6).
NINCEP	I	----	The number of intercepts for a particular radius.
NINMAR	I	----	Last (previous) number of line intersections (goes with LASTR).
PI	R	----	Pi.
R	R	feet	Current total radius point determined from RINCEP(1-4).
RINCEP (5)	R	feet	The radius at the intersection of the ray and line in question.
RINMR1-4	R	feet	Last radii points at intercepts 1-4.
RLAST	R	feet	Last total radius point.
SLOPE (500)	R	----	The slopes of the lines between data sets.
SUBST	R	----	Temperary value for the calculation of the radii.
SUMR	R	feet	The summation of the different radii values.
TEMP	R	----	Temporary variable (used in sort loops).
THETA (500)	R	radians	The angle, theta, corresponding to each x,y data set.
TLAREA	R	ft2	The total planform area.
TRANS	R	----	Temperary value for the calculation of the radii.
YESNO	C*3	----	Temp. variable for run again questions (for IFLAG = 1).

```

C
C COMMONS USED:
C   None.
C FILES USED:
C   LOGICAL UNIT      I/O  DESCRIPTION
C       7             I    X-Y data file for planform.
C NON-STANDARD CODE:
C   None.
C NOTES: Can be run as a stand alone routine if called with IFLAG
C       set = 1.
C CALLED BY:
C   NAME      DESCRIPTION
C   FINVAR    Calculates final variables which are to be used in the
C             PIE module.
C SUBROUTINES CALLED:
C   NAME      DESCRIPTION
C   None.
C ENVIRONMENT:
C   VAX, VMS, FORTRAN
C AUTHOR(S):
C   Beth Kader, Student, NASA Ames
C   Kipp E. Howard, Grad Student, San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   07/22/88  BK  -- Routine in working order.
C   08/03/88  DAW -- Added more comments to this routine.
C   08/16/88  DAW -- Still more comments/clean-up.
C   04/16/90  KEH -- Adaped routine to work with PIE module
C   07/14/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----
C   REAL ALPHA, AREAL, ARM1, ARM2, ARM3, ARM4, BKAREA, DBAR, DELTA,
C   $ FRAREA, GEOCTR, PI, R, RINCEP(5), RINMAR1, RINMAR2, RINMAR3,
C   $ RINMAR4, RLAST, SLOPE(500), SUBST, SUMR, TEMP, THETA(500),
C   $ TLAREA, TRANS, X(500), XCTR, Y(500)
C   INTEGER IND, IND2, IND3, IERROR, IFLAG, NANGLE, NINCEP, NINMAR,
C   $ NPTS
C
C   character*3 yesno
C   character*50 userfi
C
C   I/O with the user (IFLAG = 1).
C   if (iflag .eq. 1) then
C       write(6,*) ' '
C       write(6,*) 'This program is designed to work only for symmetrical'
C       write(6,*) 'airplanes -- which are probably the only kind of'
C       write(6,*) 'airplanes that you re working with, but I thought'
C       write(6,*) 'that I d tell you anyway.'
C       write(6,*) ' '
C       write(6,*) 'Enter coordinates for your aircraft in a clockwise
C $direction sequentially from the left (the nose of the aircraft).'
C       end if
3   if (iflag .eq. 1) then
C       write(6,*) ' '
C       write(6,*) 'Input filename for X and Y coordinates:'
C       read(*,5) userfi
5       format(a50)
C   endif
C

```

```

c   Set/Reset indicators.
6   ind = 0
    ind2 = 0
    ind3 = 0
    if (iflag .eq. 1) then
        write(6,*) ' '
        write(6,*) 'How many angle divisions would you like in the half
$ circle used in determining D-Bar and the area for the aircraft?
$ Areas for the front and back of the aircraft are
$ calculated using only half of the angles entered here.'
        read*, nangle
    endif
    ierror = 0
    if (nangle.lt.1) then
        if (iflag.eq.0) then
            nangle = abs(nangle)
            ierror = 3
            if (nangle.lt.1) then
                nangle = 1000
            endif
        else
            write(6,*) ' '
            write(6,*) 'Get real! (Just can not let you do it Jeffrey.)'
            go to 6
        endif
    endif
8   if (iflag.eq.1) then
        write(6,*) ' '
        write(6,*) 'Input the X-coordinate of the point at which you wish
$ D-Bar to be calculated from or the X-coordinate of the nozzle
$ location for an area value.'
        read*, xctr
    endif
9   if (iflag.eq.1) then
        write(6,*) ' '
    endif

c   Initialize variables.
c   pi = 3.14159265
    delta = pi/nangle
    dbar = 0.0
    sumr = 0.0
    areal = 0.0
    tlarea = 0.0
    frarea = 0.0
    bkarea = 0.0
    arm1 = 0.0
    arm2 = 0.0
    arm3 = 0.0
    arm4 = 0.0

c   Access user data and calculate angles and slopes.
c   IF (IFLAG .EQ. 1) THEN
        open (unit = 7, name = userfi, type = 'old')
        read (7,*) x(1), y(1)
        x(1) = x(1)
        I = 2
10    read (7, *, end = 12) x(I), y(I)

```

```

        x(I) = X(I)
        I = I + 1
        goto 10
12      npts = I-1
      END IF
C
      geoctr = (x(1)+x(npts))/2.0
C
C      Set x-position for dbar calculation equal to GEOCTR if points
C      outside of planform and/or instructed to do so.
C      if (ind2.eq.1) xctr = geoctr
C      Move input x-values to place XCTR at 0,0.
      x(1) = x(1) - xctr
C
C      Find theta's.
      do 20 I=2,npts
        x(I) = x(I) - xctr
        if (abs(x(I)).le.0.0001) then
          theta(I) = pi/2
        else if (x(I).lt.0.0) then
          theta(I) = abs(atan (y(I)/x(I) ))
        else if (x(I).gt.0.0) then
          theta(I) = pi - abs(atan (y(I)/x(I)))
        endif
        if (iflag.eq.1) then
          write(6,*) 'angle to coordinate = ',theta(I)*180.0/pi
        endif
C
C      Find slopes.
        if (x(I).eq.x(I-1)) then
          slope(I) = 9999.9
          slope of 9999.9 signifies an infinite slope (vertical
          line) .
        else
          slope(I) = (y(I)-y(I-1))/(x(I)-x(I-1))
        endif
20      continue
C
C      Catching faulty input so as to avoid inaccurate data if possible.
C      npts = I-1
      if (x(1).gt.x(npts)) then
        if (iflag.eq.1) then
          write(6,*) 'Error in data. Please enter coordinates from left
          $to right.'
          go to 3
        else
          ierror = 4
          go to 500
        endif
      else
        theta(1) = 0
      endif
      if (y(1).ne.y(npts).or.y(1).ne.0.or.y(npts).ne.0) then
        if (iflag.eq.1) then
          write(6,*) 'Please, try to follow the directions. Half an
          $ airplane on the x-axis. i.e. both your first and last y values
          $ should be 0. Thank you.'
          go to 3
        endif
      endif

```

```

        else
            ierror = 5
            go to 500
        endif
    endif
endif
if (x(npts).le.0.or.x(1).ge.0) then
    if (iflag.eq.0) then
        ierror = 2
        if (x(npts).lt.0) then
            ind = 1
        else
            ind = 2
        endif
        ind2 = 1
        rewind (7)
        goto 9
    else
        write(6,*) ' '
        write(6,*) 'Are you sure that you want your point outside of
$ the aircraft (or at the very edge)?'
        read (*,1000) yesno
        if (yesno(1:1).eq.'y') then
            write(6,*) ' '
            write(6,*) 'With this value D-Bar is inaccurate.'
            write(6,*) 'Along with the area values. An accurate
$ area measurement can be taken at the edge of the aircraft or
$ inside. While an accurate D-Bar measurement should be taken from
$ the center of gravity or thereabouts.'
            else
                write(6,*) ' '
                write(6,*) 'Would you like your point to calculate the
$ total area from the geometric center?This is probably the best
$ option if your initial point is outside the aircraft, since
$ D-Bar is invalid. '
                read (*,1000) yesno
                if (yesno(1:1).eq.'y') then
                    if (x(npts).lt.0) then
                        ind = 1
                    else
                        ind = 2
                    endif
                    ind2 = 1
                    rewind (7)
                    go to 9
                else
                    rewind (7)
                    goto 8
                endif
            endif
        endif
    endif
endif
if (iflag.eq.1) then
    write(6,*) ' '
    write(6,*) npts, ' data points received from file.'
endif
c
c Determine intersections/ # of int's/ radii.
alpha = 0.0

```

```

trans = 0.0
subst = 0.0
r = 0.0
ninmar = 1
lastr = abs(x(1))
do 70 I = 1, nangle
  nincep = 0
  alpha = alpha + delta
  if (alpha.gt.pi) then
    alpha = pi
  endif
  do 80 J = 1, npts-1
    if
$ (theta(J).lt.alpha.and.alpha.le.theta(J+1).or.theta(J+1).le.alpha
$ .and.alpha.lt.theta(J)) then
      nincep = nincep + 1
      if (slope(J+1).ge.9999.0) then
C
C      Vertical line!
      rincep(nincep) = abs(x(j)/cos(alpha))
    else
      if (slope(J+1).eq.-tan(alpha)) then
C
C      The intersection is a line (instead of a point).
      rincep(nincep) = abs((x(J) + x(J+1))/2*cos(alpha))
      if (iflag.eq.1) then
        write(6,*) 'oooooh, fun stuff!'
        Yes, I meant to leave that comment in.
      endif
    else
      if ((pi/2.0 - 0.01).le.alpha.and.alpha.le.(pi/2.0
      + 0.01)) then
$      rincep(nincep) = abs(-slope(J+1)*x(J)+y(J))
    else
      if (alpha.lt.pi/2.0) then
$      trans = (-slope(J+1)*x(J) +
$      y(J))/(-tan(alpha)-slope(J+1))
      else
$      trans = (-slope(J+1)*x(J) +
$      y(J))/(abs(tan(alpha))-slope(J+1))
      endif
      rincep(nincep) = abs(trans/cos(alpha))
    endif
  endif
endif
endif
80 continue
C
C Determining # of intercepts at each angle and the radius thereof
if (nincep.eq.1) then
  r = abs(rincep(1))
  areal = ((rlast+r)/2.0)**2.0*pi/nangle
else if
$ (nincep.eq.2.and.rincep(1).le.(rincep(2)+0.01).and.(rincep(2)-
$ 0.01).le.rincep(1)) then
  r = abs(rincep(1))
  areal = ((rlast+r)/2.0)**2.0*pi/nangle
else if (nincep.eq.2.and.rincep(1).ne.rincep(2)) then

```

```

    r = abs(rincep(2) - rincep(1))
    arm1 = abs(((rincep(1) + rinmr1))/2.0)**2.0*pi/nangle
    arm2 = abs(((rincep(2) + rinmr2)/2.0)**2.0*pi/nangle
    areal = abs(arm2 - arm1)
else if (nincep.eq.3) then
c
c    Sort radii values (three intersection points).
do 90 K = 1,2
    do 100 L = k+1,3
        if (rincep(K).gt.rincep(L)) then
            temp = rincep(L)
            rincep(L) = rincep(K)
            rincep(K) = temp
        endif
100    continue
90    continue
    if (rincep(1).eq.rincep(2)) then
        r = abs(rincep(3))
        areal = ((rlast+r)/2.0)**2.0*pi/nangle
    else
        r = abs(rincep(1) + rincep(3) - rincep(2))
        if (ninmar.eq.1.or.ninmar.eq.2) then
            areal = ((rlast+r)/2.0)**2.0*pi/nangle
        else if (ninmar.eq.3) then
            arm1 = abs(((rincep(1) + rinmr1))/2.0)**2.0*pi/nangle
            arm2 = abs(((rincep(2) + rinmr2)/2.0)**2.0*pi/nangle
            arm3 = abs(((rincep(3) + rinmr3)/2.0)**2.0*pi/nangle
            areal = arm1 - arm2 + arm3
        else if (ninmar.eq.4) then
            arm1 = abs(((rincep(1) + rinmr1))/2.0)**2.0*pi/nangle
            arm2 = abs(((rincep(2) + rinmr3)/2.0)**2.0*pi/nangle
            arm3 = abs(((rincep(3) + rinmr4)/2.0)**2.0*pi/nangle
            areal = arm1 - arm2 + arm3
        else
            areal = ((rlast+r)/2.0)**2.0*pi/nangle
        endif
    endif
else
    if (nincep.eq.4) then
c
c    Sort radii values (four intersection points).
do 110 K = 1,3
    do 120 L = K+1,4
        if (rincep(K).gt.rincep(L)) then
            temp = rincep(L)
            rincep(L) = rincep(K)
            rincep(K) = temp
        endif
120    continue
110    continue
    if (rincep(1).eq.rincep(2).and.rincep(3).eq.rincep(4)) then
        r = abs(rincep(3))
        areal = ((rlast+r)/2.0)**2.0*pi/nangle
    else if (rincep(1).eq.rincep(2)) then
        r = abs(rincep(1) + rincep(4) - rincep(3))
        if (ninmar.eq.4) then
            arm1 = abs(((rincep(1) + rinmr1))/2.0)**2.0*pi/nangle
            arm3 = abs(((rincep(3) + rinmr3)/2.0)**2.0*pi/nangle

```



```

        arm4 = abs(((rincep(4) + rinmr4)/2.0))**2.0*pi/nangle
        areal = arm1 - arm3 + arm4
    else if (ninmar.eq.3) then
        arm1 = abs(((rincep(1) + rinmr1)/2.0))**2.0*pi/nangle
        arm3 = abs(((rincep(3) + rinmr2)/2.0))**2.0*pi/nangle
        arm4 = abs(((rincep(4) + rinmr3)/2.0))**2.0*pi/nangle
        areal = arm1 - arm3 + arm4
    else if (ninmar.eq.2.or.ninmar.eq.1) then
        areal = ((rlast+r)/2.0)**2.0*pi/nangle
    else
        ind3 = 1
        r = abs(rincep(1))
        areal = ((rlast+r)/2.0)**2.0*pi/nangle
    endif
endif
endif
endif
sumr = sumr + r
C
C Calculate the total area of the aircraft (or planform).
tlarea = areal + tlarea
if (alpha.le.pi/2.0) then
    frarea = areal + frarea
else
    bkarea = areal + bkarea
endif
C
C Resetting for next go-around.
rlast = r
ninmar = nincep
if (nincep.eq.4) then
    go to 300
else if (nincep.eq.3) then
    go to 310
else if (nincep.eq.2) then
    go to 320
else if (nincep.eq.1) then
    go to 330
else
C
C Blow Up!
C NINCEP is not equal to 1,2,3, or 4
    if (nincep.eq.0 .and. iflag.eq.1) then
        write(6,*) 'trouble'
    else if (nincep.gt.4 .and. iflag.eq.1) then
        write(6,*) 'more trouble'
    else if (iflag.eq.1) then
        write(6,*) 'this statement should never print up'
    endif
    ind3 = 1
    ierror = 6
endif
C
300 rinmr4 = rincep(4)
310 rinmr3 = rincep(3)
320 rinmr2 = rincep(2)
330 rinmr1 = rincep(1)
C

```

```

70  continue
C
C  Warn user in case of possible error.
  if (ind3.eq.1.and.iflag.eq.1) then
    write(6,*) 'Possible error in D-Bar calculation from odd
$ airplane configurations or irregular placement of center
$ point. Try evaluation again except with a different
$ number of angles this time.'
  endif
C
C  D-Bar calculation.
  dbar = 2*sumr/nangle
  if (iflag.eq.1) then
    write(6,*) ' '
    write(6,*) 'D-Bar  =', dbar
    write(6,*) ' '
  endif
  if (ind.eq.1) then
    frarea = tlarea
    bkarea = 0.0
  else if (ind.eq.2) then
    bkarea = tlarea
    frarea = 0.0
  endif
  if (iflag.eq.1) then
    write(6,*) 'Total aircraft area =', tlarea
    write(6,*) 'Front aircraft area =', frarea
    write(6,*) 'Back aircraft area =', bkarea
    write(6,*) ' '
    write(6,*) 'Would you like to use the same file again?'
    read (*,1000) yesno
    if (yesno(1:1).eq.'y') then
      rewind(7)
      goto 6
    else
      close (7)
    endif
    write(6,*) 'Would you like to use a different file?'
    read (*,1000) yesno
1000  format(a3)
    if (yesno(1:1).eq.'y') then
      goto 3
    endif
  endif
  if (ind3.eq.1) then
    ierror = 6
  endif
  close (7)
500  return
end

```

Hcall

SUBROUTINE HCALL

```

C-----
C ACRONYM:  Hover CALLing routine
C          -   ----
C PURPOSE:  The purpose of HCALL is to set up all the correct variables,

```

```

C      based on the configuration, that are to be sent to the
C      HOV_GE routine. Another reason for this subroutine is all
C      subroutines that need to be called have variables with the
C      same name which need to be kept separate when using common
C      blocks to pass variables back and forth between routines.
C      LOCAL VARIABLES (in addition to the above parameters):
C      NAME      TYPE  I/O  UNITS      DESCRIPTION
C      -----
C      <list of local variables in routine>
C
C      GLOBAL VARIABLES (in addition to the above parameters and local vars):
C      -----
C      FIGPIE Common Block
C      NAME      TYPE  I/O  UNITS      DESCRIPTION
C      -----
C      B_B        R      I      Ft      Width of Body
C      B_WB       R      I      Ft      Width of Wing-Body
C      DB_B       R      I      Ft      Dbar of Body
C      DB_WB      R      I      Ft      Dbar of Wing-Body
C      DE_FR      R      I      Ft      Effective Diameter of Front & Rear
C      E_1        R      I      Ft      Half distance between adjacent
C      jets (1)
C      E_3        R      I      Ft      Half distance between adjacent
C      jets (3)
C      E_4        R      I      Ft      Half distance between adjacent
C      jets (4)
C      KR         R      I      ----    Body contour factor
C      L_WB       R      I      Ft      Length of Wing-Body
C      NUM        I      I      Ft      Total NUMBER of jets
C      PER_FR     R      I      Ft      Total perimeter of jets
C      PP_LID     R      I      ----    Ratio of perimeter enclosed by lids
C      PR_FR      R      I      ----    Jet Pressure Ratio for all jets
C      S_B        R      I      Sq Ft   Area of Body
C      S_FA1      R      I      Sq Ft   Area affected by fountain arm (1)
C      S_FA3      R      I      Sq Ft   Area affected by fountain arm (3)
C      S_FA4      R      I      Sq Ft   Area affected by fountain arm (4)
C      S_FR       R      I      Sq Ft   Total jet exit area
C      S_JP       R      I      Sq Ft   Area enclosed by Jet Pattern
C      S_LID      R      I      Sq Ft   Area enclosed by LIDS
C      S_WB       R      I      Sq Ft   Area of Wing-Body
C      SP_FA1     R      I      Sq Ft   Potential area affected by fountain
C      arm (1)
C      SP_FA3     R      I      Sq Ft   Potential area affected by fountain
C      arm (3)
C      SP_FA4     R      I      Sq Ft   Potential area affected by fountain
C      arm (4)
C      SP_JP      R      I      Sq Ft   Actual surface area within area
C      enclosed by nozzles
C      SPLY_F     R      I      Ft      SPLaY angle of Front jet
C      THA_1      R      I      deg    Half angle between jets (1)
C      THA_3      R      I      deg    Half angle between jets (3)
C      THA_4      R      I      deg    Half angle between jets (4)
C      WL_JP      R      I      ----    Width to Lenght ratio of Jet
C      Pattern
C      X_FR       R      I      Ft      Distance between Front & Rear jets
C      XCA_CG     R      I      Ft      Distance of center of area ahead of
C      Y_1        R      I      Ft      Spanwise extent of planform on
C      fountain arm (1) center line.

```

C	Y_3	R	I	Ft	Spanwise extent of planform on fountain arm (3) center line.
C	Y_4	R	I	Ft	Spanwise extent of planform on fountain arm (4) center line.
C	Y_F	R	I	Ft	Distance between Front jets
C	Y_R	R	I	Ft	Distance between Rear jets
C	YB_F	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	YP_1	R	I	Ft	Max spanwise extent of planform (1)
C	YP_3	R	I	Ft	Max spanwise extent of planform (3)
C	YP_4	R	I	Ft	Max spanwise extent of planform (4)
C	YWB_F	R	I	Ft	Lateral distance from Wingbody to Front jets (for external jets)
C	Z_W	R	I	Ft	height of wing above nozzle
C	-----				
C	HOVPIE Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	----	----	-----	-----
C	AJ	R	I	Sq Ft	Total jet exit area.
C	BF1	R	I	Ft	Half length of fountain arm (1).
C	BF3	R	I	Ft	Half length of fountain arm (3).
C	BF4	R	I	Ft	Half length of fountain arm (4).
C	BFP1	R	I	Ft	Maximum spanwise extent of fountain arm (1).
C	BFP3	R	I	Ft	Maximum spanwise extent of fountain arm (3).
C	BFP4	R	I	Ft	Maximum spanwise extent of fountain arm (4).
C	DB	R	I	Ft	Body D bar.
C	DC	R	I	Ft	Configuration D bar.
C	DE	R	I	Ft	Equivalent diameter of jets.
C	DH	R	I	Ft	Delta height of wing.
C	E	R	I	Ft	Length to width ratio of jet pattern.
C	HEIGHT	R	I	Ft	Altitude at which calculations are to be made.
C	HW	R	I	Ft	Half width of body for jets outside outside of body, else use half distance between adjacent jets.
C	KRB	R	I	Ft	Body contour factor
C	L	R	I	Ft	Length of configuration.
C	NJETS	I	I	----	Total number of jet on config.
C	PER	R	I	Ft	Total perimeter of jets (all jets)
C	PP	R	I	----	Fraction of lid perimeter enclosed
C	PR	R	I	----	Jet pressure ratio.
C	SB	R	I	Sq Ft	Planform area of body.
C	SC	R	I	Sq Ft	Area enclosed by jet pattern.
C	SCP	R	I	Sq Ft	Actual surface area inside jet pattern.
C	SFA1	R	I	Sq Ft	Area affected by fountain ares(1).
C	SFA3	R	I	Sq Ft	Area affected by fountain ares(3).
C	SFA4	R	I	Sq Ft	Area affected by fountain ares(4).
C	SFA1P	R	I	Sq Ft	Potential surface area between jets (1)
C	SFA3P	R	I	Sq Ft	Potential surface area between jets (3)
C	SFA4P	R	I	Sq Ft	Potential surface area between jets (4)

NAME	TYPE	I/O	UNITS	DESCRIPTION
SL	R	I	Sq Ft	Area enclosed by lids.
SP	R	I	Sq Ft	Planform area of configuration
SPLAY	R	I	Deg	Front jet splay angle.
THA1	R	I	Deg	Half angles between jets (1).
THA3	R	I	Deg	Half angles between jets (3).
THA4	R	I	Deg	Half angles between jets (4).
WB	R	I	Ft	Width of body.
WC	R	I	Ft	Width of configuration.
X1	R	I	Ft	Half distance between adjacent jets (1).
X3	R	I	Ft	Half distance between adjacent jets (3).
X4	R	I	Ft	Half distance between adjacent jets (4).
XCA	R	I	Ft	Distance center of area ahead of CG
YF	R	I	Ft	Distance between front jets.
YR	R	I	Ft	Distance between rear jets.

RESPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
H	I	O	----	Height number counter
HT(500)	R	O	Ft	Actual height value

FILES USED:

LOGICAL UNIT	I/O	DESCRIPTION
(none)		

COMMONS USED:

NAME	DESCRIPTION
FIGPIE	conFIGuration for Power Induced Effects - Contains all variables needed to define the configuration and other parameters of the aircraft.
HOVPIE	HOVer variables for Power Induced Effects - Contains variables which are sent to the HOV_GE subroutine.
RESPIE	RESults from all Power Induced Effects subroutines - Contains results from each subroutine along with variables for height, velocity, jet deflection angle, and angle of attack and their counters.

CALLED BY:

NAME	DESCRIPTION
COPPIE	Controls execution and coordination of Power Induced Effects Module
HOV_GE	Calculates hover lift increments for OGE and GE suckdown, and fountain effects

ROUTINES CALLED:

NAME	DESCRIPTION
HOV_GE	Calculates hover lift increments for OGE and GE suckdown, and fountain effects

NOTES: None.

REFERENCES:

1) None.

ENVIRONMENT:

FORTTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.

NON-STANDARD CODE:

```

C   ?
C   AUTHOR(S):
C     Kipp E.Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C   REVISION HISTORY:
C     DATE      INITIALS & DESCRIPTION
C     04/25/90  KEH -- Completed initial coding and debugging
C     07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C-----
#include "figpie.inc"
#include "hovpie.inc"
#include "respie.inc"
C
C Assign local HOV_GE variables to their respective global configuration
C variables.
C   Start with variables that do not change with planform configuration
HEIGHT = HT(H)
PR = PR_FR
DE = DE_FR
AJ = S_FR
PER = PER_FR
NJETS = NUM
SFA1 = S_FA1
SFA3 = S_FA3
SFA4 = S_FA4
SFA1P = SP_FA1
SFA3P = SP_FA3
SFA4P = SP_FA4
X1 = E_1
X3 = E_3
X4 = E_4
BF1 = Y_1
BF3 = Y_3
BF4 = Y_4
BFP1 = YP_1
BFP3 = YP_3
BFP4 = YP_4
THA1 = THA_1
THA3 = THA_3
THA4 = THA_4
E = 1.0/WL_JP
SC = S_JP
SCP = SP_JP
SL = S_LID
PP = PP_LID
SPLAY = SPLY_F
YF = Y_F
YR = Y_R
XCA = XCA_CG
KRB = KR
C   Assign variables that do change with planform configuration
DC = DB_WB
DB = DB_B
SP = S_WB
SB = S_B
WC = B_WB
WB = B_B
L = L_WB
DH = Z_W

```

```

C      IF (NUM .EQ. 2 .AND. (YB_F .GT. 0.0 .OR. YWB_F .GT. 0.0)) THEN
C
C          Low wing
C          IF (Z_W .LE. 0.0) THEN
C              HW = YB_F
C          High wing
C          ELSE
C              HW = YWB_F
C          END IF
C
C      ELSE
C          HW = E_1
C      END IF
C
C      Calculate Lift losses while in hover (OGE & IGE)
C      CALL HOV_GE
C
C      RETURN
C      END

```

Hov_ge

SUBROUTINE HOV_GE

```

C-----
C ACRONYM:  jet effects in HOVer due to Ground Effects.
C
C PURPOSE:  Calculation of jet induced lift loss for powered lift
C           aircraft due to jet effects in ground proximity.
C LOCAL VARIABLES (in addition to the above parameters):
C
C  NAME      TYPE  I/O  UNITS  DESCRIPTION
C  -----
C  DI         R    -    Ft      Average diameter of individual
C                               nozzles.
C  EAV        R    -    Ft      Average half distance between
C                               adjacent jets.
C  HDR        R    -    ----    aircraft Height/Effective diameter
C  HFLAG      I    -    ----    HFLAG= 1: the 2nd point tried.
C                               = 2: at least 3rd point tried.
C                               = 3: on transition line.
C                               = 4: past transition line.
C  HP         R    O    Ft      Height to which tangent line
C                               intersects dL/T=0 (Used in Hover
C                               hprime method, obviously)
C  IERROR     I    -    ----    IERROR= 1: Using lower point and
C                               possible error
C                               = 2: Past H'/De on first try.
C                               = 3: Tangent line does not
C                               intersect lower curve
C  ITRAN      I    -    ----    ITRAN = 0: Intersection of tangent
C                               line has not been found.
C                               = 1: Intersection of tangent
C                               line has been past.
C  KL         R    -    ----    Augmentation factor for LIDs
C  KLA        R    -    ----    First augmentation factor for LIDs
C  KLB        R    -    ----    Second augmentation factor for LIDs
C  KP         R    -    ----    Constant used in h' method.
C  KS         R    -    ----    Ratio of the multijet suckdown to

```

C					that for an equivalent single jet.
C	LP	R	-	----	Constant exponent used in h' method
C	LS	R	-	----	Exponent in expression for multijet
C					suckdown.
C	LTA	R	O	----	Total lift increment resulting
C					from all the fountain arms.
C	LTA1	R	O	----	Lift loss resulting from fountain
C					arm (1).
C	LTA3	R	O	----	Lift loss resulting from fountain
C					arm (3).
C	LTA4	R	O	----	Lift loss resulting from fountain
C					arm (4).
C	LTC	R	O	----	Total lift loss resulting from
C					fountain core.
C	LTCA	R	O	----	Total Lift loss resulting from
C					fountain core (Method A).
C	LTCA1	R	O	----	Lift loss from the fountain core
C					due to fountain arm (1).
C	LTCA3	R	O	----	Lift loss from the fountain core
C					due to fountain arm (3).
C	LTCA4	R	O	----	Lift loss from the fountain core
C					due to fountain arm (4).
C	LTCB	R	O	----	Total Lift loss resulting from
C					fountain core (Method B).
C	LTCB1	R	O	----	Lift loss from the fountain core
C					due to fountain arm (1).
C	LTCB3	R	O	----	Lift loss from the fountain core
C					due to fountain arm (3).
C	LTCB4	R	O	----	Lift loss from the fountain core
C					due to fountain arm (4).
C	LTF	R	O	----	Total lift loss from all fountain
C					arms.
C	LTFP	R	-	----	Lift loss from fountain arms, used
C					determine transition line.
C	LTFPL	R	-	----	Lift loss from fountain arms, used
C					determine transition line (Lower
C					line).
C	LTFPT	R	-	----	Lift loss from fountain arms, used
C					to determine transition line
C					(Transition line).
C	LTFPU	R	-	----	Lift loss from fountain arms, used
C					determine transition line. (Upper
C					line).
C	LTL	R	O	----	Total lift loss from lift
C					improvement devices.
C	LTN	R	O	----	Net lift loss for aircraft.
C	LTOG	R	O	----	Total lift resulting from suckdown
C					out of ground effect.
C	LTOGB	R	O	----	Total lift resulting from OGE
C					suckdown effect for body alone.
C	LTOGWB	R	O	----	Total lift resulting from suckdown
C					out of ground effect for wingbody.
C	LTS	R	O	----	Total lift loss resulting from
C					suckdown.
C	LTSMB	R	O	----	Lift loss resulting from suckdown
C					using the body planform and
C					multiple jets.
C	LTSSB	R	O	----	Lift loss resulting from suckdown

C					using the body planform and a single jet.
C					Lift loss resulting from suckdown
C	LTSSWB	R	O	----	using the wingbody planform and a single jet.
C					Often used parameter= $HDE/(DB/DE-1)$
C	MESS	R	-	----	Height ratio for fountain arm (1).
C	SHR1	R	-	----	Height ratio for fountain arm (3).
C	SHR3	R	-	----	Height ratio for fountain arm (4).
C	SHR4	R	-	----	Slope of temporary transition line.
C	SLOPE	R	-	----	Slope of lower line.
C	SLOPEL	R	-	----	Previous slope of temporary transition line.
C	SLOPEP	R	-	----	Slope of transition line.
C					Slope of upper line.
C	SLOPET	R	-	----	Distance to center of jet pattern when jet reaches the ground.
C	SLOPEU	R	-	----	X-intercept of current line.
C	X1C	R	-	----	Previous X-intercept of current line.
C					Previous X-coordinate of last point.
C	XINTER	R	-	----	X-coordinate of upper line.
C	XINTP	R	-	----	Previous Y-coordinate of last point.
C					Y-coordinate of Upper Line
C	XP	R	-	----	
C					
C	XUPPER	R	-	----	
C	YP	R	-	----	
C					
C	YUL	R	-	----	
C	GLOBAL VARIABLES (in addition to the above parameters and local vars):				
C	-----				
C	HOVPIE Common Block				
C					
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----	-----	-----	-----	-----
C	AJ	R	I	Sq Ft	Total jet exit area.
C	BF1	R	I	Ft	Half length of fountain arm (1).
C	BF3	R	I	Ft	Half length of fountain arm (3).
C	BF4	R	I	Ft	Half length of fountain arm (4).
C	BFP1	R	I	Ft	Maximum spanwise extent of fountain arm (1).
C					Maximum spanwise extent of fountain arm (3).
C	BFP3	R	I	Ft	Maximum spanwise extent of fountain arm (4).
C	BFP4	R	I	Ft	Body D bar.
C					Configuration D bar.
C	DB	R	I	Ft	Equivalent diameter of jets.
C	DC	R	I	Ft	Delta height of wing.
C	DE	R	I	Ft	Length to width ratio of jet pattern.
C	DH	R	I	Ft	Altitude at which calculations are to be made.
C	E	R	I	Ft	Half width of body for jets outside outside of body, else use half distance between adjacent jets.
C					Body contour factor
C	HEIGHT	R	I	Ft	Length of configuration.
C					Total number of jet on config.
C	HW	R	I	Ft	Total perimeter of jets (all jets)
C					Fraction of lid perimeter enclosed
C	KRB	R	I	Ft	
C	L	R	I	Ft	
C	NJETS	I	I	----	
C	PER	R	I	Ft	
C	PP	R	I	----	

C	PR	R	I	----	Jet pressure ratio.
C	SB	R	I	Sq Ft	Planform area of body.
C	SC	R	I	Sq Ft	Area enclosed by jet pattern.
C	SCP	R	I	Sq Ft	Actual surface area inside jet pattern.
C					
C	SFA1	R	I	Sq Ft	Area affected by fountain ares(1).
C	SFA3	R	I	Sq Ft	Area affected by fountain ares(3).
C	SFA4	R	I	Sq Ft	Area affected by fountain ares(4).
C	SFA1P	R	I	Sq Ft	Potential surface area between jets (1)
C					
C	SFA3P	R	I	Sq Ft	Potential surface area between jets (3)
C					
C	SFA4P	R	I	Sq Ft	Potential surface area between jets (4)
C					
C	SL	R	I	Sq Ft	Area enclosed by lids.
C	SP	R	I	Sq Ft	Planform area of configuration
C	SPLAY	R	I	Deg	Front jet splay angle.
C	THA1	R	I	Deg	Half angles between jets (1).
C	THA3	R	I	Deg	Half angles between jets (3).
C	THA4	R	I	Deg	Half angles between jets (4).
C	WB	R	I	Ft	Width of body.
C	WC	R	I	Ft	Width of configuration.
C	X1	R	I	Ft	Half distance between adjacent jets (1).
C					
C	X3	R	I	Ft	Half distance between adjacent jets (3).
C					
C	X4	R	I	Ft	Half distance between adjacent jets (4).
C					
C	XCA	R	I	Ft	Distance center of area ahead of CG
C	YF	R	I	Ft	Distance between front jets.
C	YR	R	I	Ft	Distance between rear jets.

C RESPIE Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	H	I	O	----	Height number counter
C	H_HPRI	R	O	Ft	Height to which tangent line intersects dL/T=0 (Used in Hover hprime method, obviously)
C					
C	H_LT(500)	R	O	----	Total lift loss calculated while in hover.
C					
C	H_LTF(500)	R	O	----	Lift gain due to fountain effects while in hover.
C					
C	H_LTL(500)	R	O	----	Lift gain due to the addition of LIDs while in hover.
C					
C	H_LTOG	R	O	----	Lift loss OGE effect while in hover
C	H_LTS(500)	R	O	----	Lift loss due to suckdown while in hover.

C FILES USED:

C	LOGICAL UNIT	I/O	DESCRIPTION
C	None.		

C COMMONS USED:

C	NAME	DESCRIPTION
C	HOVPIE	HOVer variables for Power Induced Effects - Contains variables which are sent to the HOV_GE subroutine.
C	RESPIE	RESults from all POver Induced Effects subroutines -

```

C          Contains results from each subroutine along with
C          variables for height, velocity, jet deflection angle,
C          and angle of attack and their counters.
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   HCALL     Isolates and controls execution of HOV_GE
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   None.
C NOTES: None.
C REFERENCES:
C   1) Kuhn, R.E. "An Engineering Method of Estimating the Induced Lift
C       on V/STOL Aircraft Hovering In and Out of Ground Effect." V/STOL
C       Consultant. NADC-80246-60. pp. January, 1981.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Douglas A. Wardwell (DAW), NASA Ames Research Center
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   11/28/89  DAW -- FORTRAN version of Dick Kuhn's basic program.
C   04/22/90  KEH -- Modifications and corrections of routine
C   07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "hovpie.inc"
C #include "respie.inc"
C
C *** Calculation variables.
C   INTEGER IERROR, I, HFLAG, ITRAN
C   REAL LTS, LTN, LTA, LTA1, LTA3, LTA4, LTCA, TEMP,
C   $ LTCA1, LTCA3, LTCA4, LTCB, LTCB1, LTCB3, LTCB4, SHR1,
C   $ SHR3, SHR4, KCA, KCB, KLA, KLB, KL, GS, KS, LS,
C   $ LTC, LTF, LTL, SLOPE, SLOPEL, SLOPEU, YP, XP, KP, LP,
C   $ LTFP, LTFPL, LTFPU, LTLF, XINTER, YUL, YTRAN, LTFPT, SLOPET,
C   $ LTOGB, LTOGBW, LTOG, MESS, LTSMB, LTSSB, LTSSWB,
C   $ LTSMB, EAV, DI, X1C, HDA, HP, YFC, SCC, EC, HPDE, MTD, LCA, LCB
C
C   SAVE LTFPT, SLOPET, XINTER, XP, YP, YTRAN, HFLAG, ITRAN
C
C   IF (H .EQ. 1) THEN
C     HFLAG = 0
C     ITRAN = 0
C   END IF
C
C   IF (E .LT. 1.0) E = 1.0 / E
C
C *** Estimated HOVER Suckdown and Fountain Lift (?) using the
C *** Basic method --  $X/D > 3.0$ , or h' method --  $X/D \leq 3.0$ 
C   IF (DH .NE. 0.0) THEN
C     High Wing configuration
C     LTOGB = -0.00022*SQRT(SB/AJ)*(PR**-0.64)*(PER/DE)**1.58
C   ELSE
C     Low Wing configuration

```

```

      LTOGB = -0.00022*SQRT(SP/AJ)*(PR**-0.64)*(PER/DE)**1.58
END IF
C
      LTOGBW = -0.00022*SQRT(SP/AJ)*(PR**-0.64)*(PER/DE)**1.58
      LTOG = LTOGB + (LTOGBW-LTOGB)*(1.0-0.4*SQRT(DH/DE))
      LTF = 0.0
      LTL = 0.0
      LTN = 0.0
      IERROR = 0
      HDR = HEIGHT/DE
C
C      Lift loss due to suckdown
      IF (DH .NE. 0.0) THEN
C      High wing configuration
        MESS = (HDR)/(DB/DE-1.0)
C
        IF (NJETS .EQ. 1) THEN
          KS = 1.0
        ELSE
          LS = -1.7*((WB/L)*(SB/(WB*L))**0.36)**1.38
          KS = 4.5*(MESS)**0.25 * (1.0-((HDR)/
$          (0.08*(DB/DE)*(WB/L))**LS)
        END IF
C
      IF (PR .GE. 2.0) THEN
C      Assume PR = 2.0
        LTSMB = KS * (-0.015) * (MESS**-1.96)
        LTSSB = -0.015*(((HEIGHT+DH)/DE)/(DB/DE-1.0))**-1.96
        LTSSWB = -0.015*(((HEIGHT+DH)/DE)/(DC/DE-1.0))**-1.96
      ELSE
        LTSMB = KS * (-0.015) * (MESS**-(2.2-.24*(PR-1.0)))
        LTSSB = -0.015*(((HEIGHT+DH)/DE)/(DB/DE-1.0))
$        **-(2.2-.24*(PR-1.0))
        LTSSWB = -0.015*(((HEIGHT+DH)/DE)/(DC/DE-1.0))
$        **-(2.2-.24*(PR-1.0))
      END IF
C
      LTS = LTSMB+LTSSWB-LTSSB
    ELSE
C      Low wing configuration
      MESS = (HDR)/(DC/DE-1.0)
C
      IF (NJETS .EQ. 1) THEN
        KS = 1.0
      ELSE
        LS = -1.7*((WC/L)*(SP/(WC*L))**0.36)**1.38
        KS = 4.5*(MESS)**0.25 * (1.0-((HDR)/
$        (0.08*(DC/DE)*(WC/L))**LS)
      END IF
C
      IF (PR .GE. 2.0) THEN
        LTS = KS * (-0.015)*MESS**-1.96
      ELSE
        LTS = KS * (-0.015) * (MESS**-(2.2-.24*(PR-1.0)))
      END IF
C
    ENDIF
C

```

```

DI = DE/SQRT(REAL(NJETS))
EAV = (2*X1+X3+X4)/NJETS
IF (NJETS .EQ. 2) EAV = X1
C
C
*** Fountain Effects Calculation.
IF (EAV/DI .GT. 3.0) THEN
  IF (NJETS .EQ. 2) THEN
C
C
    *** Jet Fountain -- 2 jets (Basic method).
    IF (SPLAY .GT. 0) THEN
      X1C = X1-HEIGHT*TAN(SPLAY/57.296)
      IF (X1C .LT. 0.0) X1C=0.0
      SFA1 = SFA1*X1C/X1
      SC = SC*X1C/X1
C
      IF (SFA1 .EQ. 0.0) THEN
        LTA1 = 0.0
        GO TO 1210
      ELSE
        X1=X1C
      END IF
C
    END IF
C
    SHR1 = X1/(X1+HEIGHT)
    LTA1 = (2.0/NJETS*((BFP1 * SFA1)/(X1 * SFA1P))**0.835) *
      SHR1**2.0*BF1/SQRT(BF1*BF1+(X1+HEIGHT)**2.0)
$
1210 LTF = LTA1
    LTN = LTOG+LTS+LTF
    HP = 1.0
    GO TO 2330
  ELSE IF (NJETS .GT. 2) THEN
C
C
    *** Jet Fountain -- more than 2 jets (Basic method).
    *** Fountain Arms.
    SHR1 = X1/(X1+HEIGHT)
    SHR3 = X3/(X3+HEIGHT)
    LTA1 = (2.0/NJETS*((BFP1 * SFA1)/(X1 * SFA1P))**0.835) *
      SHR1**2.0*BF1/SQRT(BF1*BF1+(X1+HEIGHT)**2.0)
$
    LTA3 = (2.0/NJETS*((BFP3 * SFA3)/(X3 * SFA3P))**0.835) *
      SHR3**2.0*BF3/SQRT(BF3*BF3+(X3+HEIGHT)**2.0)
$
C
C
    In the case of four jets
    IF (NJETS .GT. 3) THEN
      SHR4 = X4/(X4+HEIGHT)
      LTA4 = (2.0/NJETS*((BFP4 * SFA4)/(X4 * SFA4P))**0.835) *
$
        SHR4**2.0*BF4/SQRT(BF4*BF4+(X4+HEIGHT)**2.0)
    ELSE
      SHR4 = 0.0
      LTA4 = 0.0
    ENDIF
C
    LTA = ((2.0*LTA1+LTA3+LTA4)/2.0)*0.7*
$
      SQRT((HDR)/(DB/DE-1))
C
    *** Fountain Core.
    KCA = 0.12*NJETS*((DB/DE)*(WB/L)*E**0.25)*
$
      DE/SQRT(SC)
    LCA = 2.5
    LTCA1 = KCA*(SHR1)**LCA*COS(THA1)

```

```

LTCA3 = KCA*(SHR3)**LCA*COS(THA3)
LTCA4 = KCA*(SHR4)**LCA*COS(THA4)
LTCA = (2.0*LTCA1+LTCA3+LTCA4)
KCB = 0.31*NJETS*(DB/DE)**0.35*(WB/L)**0.65*
$      (SCP/SC)**0.5*((E*DE)/SQRT(SC))**1.8
LCB = NJETS*E*DE/SQRT(SC)
LTCB1 = KCB*(SHR1)**LCB*COS(THA1)
LTCB3 = KCB*(SHR3)**LCB*COS(THA3)
LTCB4 = KCB*(SHR4)**LCB*COS(THA4)
LTCB = (2.0*LTCB1+LTCB3+LTCB4)

C      IF (LTCA .GT. LTCB) THEN
          LTC = LTCA
        ELSE
          LTC = LTCB
        ENDIF

C      LTF = LTA + LTC
    ENDIF

C      HP = 1.0
    ELSE IF (NJETS .GT. 1) THEN
C      *** h' method for fountain effects.
C
C      IF (NJETS .EQ. 2) THEN
C      *** 2 jet configuration.
C      *** Use a pressure ratio (PR) of 2.0 if PR is greater
C      than two.
C
C      IF (PR .GE. 2.0) THEN
          HP = 3.6*(HW/DI)**.62*SQRT(2.0)*DE
        ELSE
          HP = 3.6*(HW/DI)**.62*SQRT(PR)*DE
        END IF

C      For 2 jet side-by-side use 2*E instead of BF1 since
C      method was developed for longitudinal jets not lateral
C      jets.
C      IF (YF .EQ. 0.0 .AND. YR .EQ. 0.0) THEN
          KP = 0.084*(EAV/DI)**.39 *
$          ((BF1/DI)*(SFA1/SFA1P))**1.1
        ELSE
          KP = .084*(EAV/DI)**.39 *
$          ((2.*X1/DI)*(SFA1/SFA1P))**1.1
        END IF

C      LP = -1.35*(HW/X1)
    ELSE

C      *** more than 2 jet configuration.
C      *** Use a pressure ratio (PR) of 2.0 if PR is greater
C      than two.
C      IF (PR .GE. 2.0) THEN
          HP = 2.*SQRT(EAV/DI)*SQRT(2.0)*DE
        ELSE
          HP = 2.*SQRT(EAV/DI)*SQRT(PR)*DE
        END IF

```

```

      KP = 4.4*(SQRT(SC)*DI/(DE*2.*EAV))**3. *
      (DB*WB/(DE*L))**.9/E
$      LP = -2.4*(DB*WB/(DE*L))**.4/(EAV/DI*SQRT(E))
ENDIF

C      SLOPEL = 0.033*(DB/DE*WB/L)
      LTFPU = KP*(HDR)**LP
      LTFPL = SLOPEL/(HDR)

C      IF ((HDR) .LT. HP/DE .OR. HFLAG .LT. 3) THEN
C
C          IF (HFLAG .EQ. 0) THEN
C              *** Use 1st point from upper curve.
C              LTFP = LTFPU
C              HFLAG = 1
C          ELSE IF (HFLAG .LT. 3) THEN
C              *** HFLAG = 1: the 2nd point tried.
C              *** HFLAG = 2: at least the 3rd point tried.
C              SLOPEP = SLOPE
C              SLOPE = (YP-LTFPU)/(XP-(HDR))
C
C              IF (SLOPE .GE. 0.0) THEN
C                  *** Use the lower point and give possible error
C                  *** message.
C                  HFLAG = 4
C                  IERROR = 1
C              ELSE
C                  *** The x-intercept (h/d) point.
C                  XINTP = XINTER
C                  XINTER = (HDR) - 1.0/SLOPE*LTFPU
C
C                  IF (XINTER .GT. HP/DE) THEN
C
C                      IF (HFLAG .EQ. 2) THEN
C                          *** Use previous line and current line to
C                          *** find the tangent line.
C                          LTFPT = YP - (YP-LTFPU)*(XINTP-HP/DE)/
$                          (XINTP-XINTER)
$                          XUPPER = XP - (XP-HDR)*(XINTP-HP/DE)/
$                          (XINTP-XINTER)
C                          SLOPET = -LTFPT/(HP/DE-XUPPER)
C                          HFLAG = 3
C                          XINTER = HP/DE
C                      ELSE
C                          *** Draw line from point to HP/DE and use
C                          *** if intersection < HP.
C                          SLOPET = (LTFP-0.0)/((HDR)-HP/DE)
C                          XINTER = HP/DE
C                          HFLAG = 3
C                          IERROR = 2
C                      ENDIF
C
C                  ELSE IF (XINTER .EQ. HP/DE) THEN
C                      *** Set up the transition line.
C                      SLOPET = SLOPE
C                      HFLAG = 3
C                  ELSE IF (XINTER .LT. HP/DE) THEN
C                      ***** Use upper line.

```

```

      LTFP = LTFPU
    ENDIF
C
    ENDIF
C
    IF (HFLAG .EQ. 1) HFLAG = 2
  ENDIF
C
    YP = LTFPU
    XP = (HDR)
  ENDIF
C
  IF (HFLAG .EQ. 3 .AND. (HDR) .LT. HP/DE) THEN
C
    *** Use the Transition Tangent line.
C
    IF (ITRAN .EQ. 0) THEN
      YUL = LTFPT
C
      IF ((HP/DE*SLOPET)**2.0 .LT.
$      ABS(4.0*SLOPEL*SLOPET)) THEN
C
      *** Tangent line DOES NOT intersect the lower
C
      *** curve.
      LTFP = LTFPL
      HFLAG = 4
      IERROR = 3
      ELSE
C
      *** Find the intersection of the tangent line and
C
      *** lower curve.
      YTRAN = (-(HP/DE)*SLOPET -
$      Sqrt((HP/DE*SLOPET)**2.0 +
$      4.0*SLOPEL*SLOPET))/2.0
      ITRAN = 1
    ENDIF
C
  ENDIF
C
  LTFP = SLOPET * ((HDR) - XINTER)
C
  IF (LTFP .LE. YTRAN) THEN
C
    *** Past the intersection of the transition line and
C
    *** the lower curve--now use lower curve.
    LTFP = LTFPL
    HFLAG = 4
  ENDIF
C
  ELSE IF ((HDR) .GE. HP/DE .OR. HFLAG .EQ. 4) THEN
C
    *** Use lower curve.
    LTFP = LTFPL
    HFLAG = 4
  ENDIF
C
  LTF = LTFP
  ELSE
C
    For the single jet case
    LTF = 0.0
  END IF
C*****
C    *** LID's

```



```

      IF (NJETS .GT. 2) THEN
C
      IF (SL .GT. 0.0 .AND. SCP .GT. 0.0) THEN
        KLA = 1.25*PP*(SL/SCP)*(DE/DB)**0.44*SQRT(1/E)
        KLB = 0.22*(HEIGHT/SQRT(SL))*E**2.0*(SC/SCP)
C
        IF (KLA .LT. KLB) THEN
          KL = KLA
        ELSE
          KL = KLB
        ENDIF
C
        LTL = LTF*KL
      ENDIF
C
    ENDIF
C
2330 CONTINUE
C
    *** Output ***
    LTF = LTF*KRB
    LTN = LTOG+LTS+LTF+LTL
    MTD = LTS*XCA/DE
C
    Assign lift losses calculated to the appropriate lift loss
C
    result variable
    H_LTOG = LTOG
    H_LTS(H) = LTS
    H_LTF(H) = LTF
    H_LTL(H) = LTL
    H_LT(H) = LTN
    H_HPRI = HP
C
    RETURN
  END

```

Sfcall

```

SUBROUTINE SFCALL
C-----
C ACRONYM:  stolsf CALLing routine.
C
C PURPOSE:  The purpose of SFCALL is to set up all correct variables,
C           based on the configuration, that are to be sent to the
C           STOLSf routine.  Another reason for this subroutine is all
C           the subroutines that need to be called have variables with
C           the same name which need to be kept separate when using
C           common blocks to pass the variables back and forth between
C           routines.
C
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   (none)
C
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C-----
C FIGPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   DB_B      R     I    Ft         Dbar of Body
C   DB_WB     R     I    Ft         Dbar of Wing-Body

```

C	DE_F	R	I	Ft	Effective Diameter of Front jets
C	DE_FR	R	I	Ft	Effective Diameter of Front & Rear nozzles
C					
C	KB	R	I	----	Boundary layer factor
C	NUM	I	I	Ft	Total NUMBER of jets
C	NUM_F	I	I	----	NUMBER of Front jets
C	NUM_R	I	I	----	NUMBER of Rear jets
C	PR_FR	R	I	----	Jet Pressure Ratio for all jets
C	Q_FR	R	I	lb/sq ft	Dynamic pressure for Front and Rear nozzles
C					
C	S_FR	R	I	Sq Ft	Total jet exit area
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	Y_F	R	I	Ft	Distance between Front jets
C	Z_W	R	I	Ft	height of wing above nozzle
C					
C	PIEFLAG Common Block				DESCRIPTION
C	NAME	TYPE	I/O	UNITS	
C	-----	---	---	-----	-----
C	FLGRCS	L	O	----	Flag which signals if there is an RCS on the configuration
C					(TRUE - RCS, FALSE - No RCS)
C	HDEOUT	L	I	----	Signals when to output tables based on height or height/De
C					TRUE - Print based on Height/DE,
C					FALSE - Print based on height
C	PRTFLG	L	I	----	Signals when to output to screen.
C	RCSFLG	L	O	----	Flag which identifies if RCS lift loss has calculation exceeded the area ratio (Swing / Ajet > 7000)
C					TRUE - Exceeded
C					FALSE - Not Exceeded
C	TYPE_F	C*4	O	----	Discription of type of front nozzle (CIRCular, OVAL, RECTangular)
C	TYPE_R	C*4	O	----	Same as TYPE_F but for rear nozzles
C	VEOUT	L	I	----	Signals when to output tables based on VE
C					TRUE - Print Based on VE
C					FALSE - Print Based on Velocity
C	WBFLAG	L	O	----	Identifies when WingBody planform has been enter by the user. Used to determine when to use wingbody in calculations.
C					
C	PIESF Common Block				DESCRIPTION
C	NAME	TYPE	I/O	UNITS	
C	-----	---	---	-----	-----
C	AJ	R	I	Sq Ft	Area of front and rear nozzles. -
C	DC	R	I	Ft	Dbar of the configuration used (Based on the wing height).
C					
C	DF	R	I	Ft	Diameter of individual front jet.
C	DE	R	I	Ft	Equivalent diameter of all jets.
C	DH	R	I	Ft	Wing height above nozzles
C	HEIGHT	R	I	Ft	Altitude at which calculations are to be made.
C					
C	HLTF	R	I	----	Hover lift gain due to fountain effects
C					

NAME	TYPE	I/O	UNITS	DESCRIPTION
HLTL	R	I	----	Lift gain due to the addition of LIDs while in hover.
HPRIME	R	I	----	H' calculated from HOVPIE
KBL	R	I	----	Boundry layer factor
LP	R	I	----	Distance between front and rear nozzles.
LTS	R	I	----	Hover lift loss due to suckdown.
N	I	I	----	Total number of nozzles.
NF	I	I	----	Total number of front nozzles.
PR	R	I	----	Pressure ratio of nozzles.
Q	R	I	lb/Sq Ft	Average dynamic pressure of nozzles
WL	R	I	----	Average width to length of front and rear nozzles
YF	R	I	----	Distance between front nozzles.

RESPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
D	I	O	----	Nozzle deflection angle number
DFL(500)	R	I	deg	Actual nozzle deflection angle
H	I	O	----	Height number counter
H_HPRI	R	O	Ft	Height to which tangent line
H_LTF(500)	R	O	----	Lift gain due to fountain effects while in hover.
H_LTL(500)	R	O	----	Lift gain due to the addition of LIDs while in hover.
H_LTS(500)	R	O	----	Lift loss due to suckdown while in hover.
HT(500)	R	O	Ft	Actual height value
V	I	O	----	Velocity number counter
VO(500)	R	O	----	Actual velocity value

FILES USED:

LOGICAL UNIT	I/O	DESCRIPTION
(none)		

COMMONS USED:

NAME	DESCRIPTION
FIGPIE	conFIGuration for Power Induced Effects - Contains all variables needed to define the configuration and other parameters of the aircraft.
PIEFLAGS	Power Induced Effects FLAGS - Contains variables which help keep track of the configuration of the aircraft.
PIESF	Power Induced Effects for stolsf - Contains variables which are sent to the STOLSf subroutine.
RESPIE	RESults from all POWER Induced Effects subroutines - Contains results from each subroutine along with variables for height, velocity, jet deflection angle, and angle of attack and their counters.

CALLED BY:

NAME	DESCRIPTION
COPPIE	Controls execution and coordination of Power Induced Effects Module

ROUTINES CALLED:

NAME	DESCRIPTION
STOLSf	Calculates suckdown and fountain lift increments while

```

C          in STOL flight
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE          INITIALS & DESCRIPTION
C   05/01/90    KEH -- Completed initial coding and debugging.
C   07/13/91    KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "figpie.inc"
C #include "pieflag.inc"
C #include "piesf.inc"
C #include "respie.inc"
C
C Assign local STOLSF variables to their respective global configuration
C variables.
C   HEIGHT = HT(H)
C   PR = PR_FR
C   DE = DE_FR
C   AJ = S_FR
C   DH = Z_W
C   N = NUM
C   NF = NUM_F
C   YF = Y_F
C   LP = X_FR
C
C   IF (NUM_R .NE. 0) THEN
C     WL = (WL_F + WL_R) / 2.0
C   ELSE
C     WL = WL_F
C   END IF
C
C   KBL = KB
C   DF = D_F
C   Q = Q_FR
C   HPRIME = H_HPRI
C   LTS = H_LTS(H)
C   HLTF = H_LTF(H)
C   HLTL = H_LTL(H)
C
C   IF (Z_W .NE. 0.0) THEN
C     DC = DB_WB
C
C   ELSE
C     DC = DB_B
C   END IF
C
C   CALL STOLSF
C
C   RETURN
C   END
C   SUBROUTINE STOLSF

```

C-----
 C ACRONYM: STOL operation - estimation of Suckdown and Fountain effects.
 C-----

C PURPOSE: The purpose of STOLSF is to calculate the suckdown and
 C fountain effects while in the STOL configuration.

C LOCAL VARIABLES (in addition to the above parameters):

NAME	TYPE	I/O	UNITS	DESCRIPTION
HD	R	-	----	Aircraft height/front nozzle diam.
HF	R	-	Ft	Height where fountain effects are diminish to zero.
KH	R	-	----	Wall jet placement factor
KX	R	-	----	Longitudinal adjustment factor
KY	R	-	----	Lateral adjustment factor
LTF	R	-	----	Storage variable for fountain lift increment
LTN	R	-	----	Storage variable for net lift increment
LTSS	R	-	----	Storage variable for suckdown lift increment
VE	R	-	----	Ratio of freestream dynamic pressure to jet dynamic pressure.
XD	R	-	----	Forward extent of wall jet in units of jet diameters

C GLOBAL VARIABLES (in addition to local variables):

C-----
 C PIESF Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
AJ	R	I	Sq Ft	Area of front and rear nozzles.
DC	R	I	Ft	Dbar of the configuration used (Based on the wing height).
DF	R	I	Ft	Diameter of individual front jet.
DE	R	I	Ft	Equivalent diameter of all jets.
DH	R	I	Ft	Wing height above nozzles
HEIGHT	R	I	Ft	Altitude at which calculations are to be made.
HLTF	R	I	----	Hover lift gain due to fountain effects
HLTL	R	I	----	Lift gain due to the addition of LIDs while in hover.
HPRIME	R	I	----	H' calculated from HOVPIE
KBL	R	I	----	Boundary layer factor
LP	R	I	----	Distance between front and rear nozzles.
LTS	R	I	----	Hover lift loss due to suckdown.
N	I	I	----	Total number of nozzles.
NF	I	I	----	Total number of front nozzles.
PR	R	I	----	Pressure ratio of nozzles.
Q	R	I	lb/Sq Ft	Average dynamic pressure of nozzles
WL	R	I	----	Average width to length of front and rear nozzles
YF	R	I	----	Distance between front nozzles.

C-----
 C RESPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
D	I	O	----	Nozzle deflection angle number

C	DFL(500)	R	I	deg	Actual nozzle deflection angle
C	H	I	O	----	Height number counter
C	H_HPRI	R	O	Ft	Height to which tangent line
C	H_LTF(500)	R	O	----	Lift gain due to fountain effects
C					while in hover.
C	H_LTL(500)	R	O	----	Lift gain due to the addition of
C					LIDs while in hover.
C	H_LTS(500)	R	O	----	Lift loss due to suckdown while in
C					hover.
C	HT(500)	R	O	Ft	Actual height value
C	V	I	O	----	Velocity number counter
C	VO(500)	R	O	----	Actual velocity value

C FILES USED:

C	LOGICAL UNIT	I/O	DESCRIPTION
C	-----	---	-----
C	70	O	Direct access file for storage of lift increment
C			due to suckdown
C	71	O	Direct access file for storage of lift increment
C			due to fountain

C COMMONS USED:

C	NAME	DESCRIPTION
C	-----	-----
C	PIESF	Power Induced Effects for stolsf - Contains
C		variables which are sent to the STOLSf subroutine.
C	RESPIE	RESULTS from all Power Induced Effects subroutines -
C		Contains results from each subroutine along with
C		variables for height, velocity, jet deflection angle,
C		and angle of attack and their counters.

C CALLED BY:

C	NAME	DESCRIPTION
C	-----	-----
C	SFCALL	Isolates and controls execution of STOLSf

C ROUTINES CALLED:

C	NAME	DESCRIPTION
C	-----	-----
C	(none)	

C NOTES: None.

C REFERENCES:

C 1) None.

C ENVIRONMENT:

C FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.

C NON-STANDARD CODE:

C ?

C AUTHOR(S):

C Kipp E.Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett

C REVISION HISTORY:

C DATE INITIALS & DESCRIPTION

C 05/10/90 KEH -- Initial completion of coding.

C 07/13/91 KEH -- Eliminated extraneous spacing and fixed comments

C #include "piesf.inc"

C #include "respie.inc"

C

C Local variable declarations

REAL VE, LTSS, KH, KY, KX, HD, XD, HF, LTF, LTN

C

VE=SQRT((0.00119*(1.69*VO(V))**2.0)/Q)

C Suckdown Calculations

```

C      HD=HEIGHT/DF
C
C      IF (VE .EQ. 0.) THEN
C          KH = 1.
C      ELSE IF (NF .EQ. 2) THEN
C
C          (AFWAL-TR-87-3019 Volume II, 14-6)
C          Make calculations for jets that are side-by-side.
C          IF ((YF/DF) .GE. 4.25 .OR. NF .EQ. 1) THEN
C              KY = 1.0
C          ELSE
C              KY = 1. + 6.2 * VE * (1. - (YF/DF)/4.25)
C          END IF
C
C          Calculate forward extent of jet flow.
C          XD=HD*TAN((DFL(D)-90.)/57.296)+(KBL*(.75/VE)-1.75*WL**.3*
$          VE**2.25*(1.-SIN((DFL(D)-90.)/57.296))*HD**2.5)*
$          (DFL(D)/90.))**2.*KY
C          KH = (XD + (LP/2.)/DF) * DF/DC + .5
C
C          IF (KH .GE. 1.) THEN
C              KH = 1.0
C          ELSE IF (KH .LT. 0.0) THEN
C              KH = 0.0
C          END IF
C
C          ELSE IF (NF .EQ. 1) THEN
C
C              (AFWAL-TR-87-3019 Volume II, 14-5)
C              Make calculations for tandem jets.
C              IF ((LP/2.)/DF .LT. 2.12) THEN
C                  KX = 1. + .64 * (1. - (LP/2.)/DF)/2.12)
C              ELSE IF ((LP/2.)/DF .GE. 2.12 .OR.
$              (LP/2.)/DF .EQ. 0.) THEN
C                  KX = 1.
C              END IF
C
C              Calculate forward extent of jet wake
C              XD=HD*TAN((DFL(D)-90.)/57.296)+(KBL*(.75/VE)-1.75*WL**.3*
$              VE**2.25*(1.-SIN((DFL(D)-90.)/57.296))*(HD/KX)**2.5)*
$              (DFL(D)/90.))**2.
C              KH = (XD + (LP/2.)/DF) * DF/DC + .5
C
C              IF (KH.GE.1.) THEN
C                  KH=1.0
C              ELSE IF (KH .LT. 0.0) THEN
C                  KH=0.0
C              END IF
C
C          END IF
C
C          LTSS=LTS*KH*(SIN(DFL(D)/57.296))**2
C
C      Fountain lift calculations
C      IF (N.LT.2)THEN
C          LTF=0.
C      ELSE
C

```

```

      IF (VE.EQ.0.) THEN
        HF=HPRIME
      ELSE
        HF=DE*2.5/SQRT(VE)
      ENDIF
C
      IF (HEIGHT.GT.HF .AND. VE .NE. 0.0) THEN
        LTF=0.
      ELSE IF (HF.GT.HPRIME) THEN
        HF=HPRIME
        LTF=(HLTF+HLTL)*(HF/HPRIME)**0.5*KH
        &      *(SIN(DFL(D)/57.296))**2.0
      ELSE
        LTF=(HLTF+HLTL)*(HF/HPRIME)**0.5*KH
        &      *(SIN(DFL(D)/57.296))**2.0
      END IF
C
      IF (LTF.LT.0.) LTF=0.0
      END IF
C
C Out-of-Ground-Effect Term
C The following has been commented out because of the
C recommendation of Richard Kuhn.
C-----
C This section must be included because the OGE lift loss
C does vary with velocity and deflection angle.
C
C      HLTOGE(D) = H_LTOG * (SIN(DFL(D) / 57.296))**2
C-----
C      HLTOGE(D) = H_LTOG
C
C OUTPUT
C      LTN = LTSS + LTF
C      HDR=HEIGHT/DE
C Assign lift losses calculated to the appropriate lift loss
C result variable
C      REC3 = LH * LV * (D - 1) + LH * (V - 1) + H
C      WRITE (70,100,REC=REC3) LTSS
C      WRITE (71,100,REC=REC3) LTF
C
C      RETURN
C Format statements
100  FORMAT(F11.5)
      END

```

Rcscal

```

SUBROUTINE RCSCAL
C-----
C ACRONYM: Reaction Control System CALL routine (power induced effects)
C
C PURPOSE: RCSCAL sets up all the correct variables, based on the
C          configuration, that are sent to the RCSCAL routine.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   (none)
C GLOBAL VARIABLES (in addition to the above parameters and local vars):

```



```

C -----
C FIGPIE Common Block
C  NAME      TYPE  I/O  UNITS      DESCRIPTION
C -----
C  B_W        R    I    Ft        Width of Wing (Wing span)
C  D_RCS       R    I    Ft        Diameter of Roll RCS nozzle
C  PR_RCS      R    I    ----      Roll RCS Pressure Ratio
C  Q_RCS       R    I    lb/sq ft   Dynamic pressure for roll RCS jets
C  S_W        R    I    Sq Ft      Area of Wing
C  T_RCS       R    I    lb        Thrust of roll RCS nozzle
C  X_RCS       R    I    Ft        Distance of roll RCS nozzle ahead
C                                     of wing trailing edge
C  Y_RCS       R    I    Ft        Distance of roll RCS nozzle in from
C                                     wingtip
C -----
C PIEFLAG Common Block
C  NAME      TYPE  I/O  UNITS      DESCRIPTION
C -----
C  FLGRCS     L    O    ----      Flag which signals if there is an
C                                     RCS on the configuration
C                                     (TRUE - RCS, FALSE - No RCS)
C -----
C PIERCS Common Block
C  NAME      TYPE  I/O  UNITS      DESCRIPTION
C -----
C  B          R    I    feet       Wing span
C  DRCS       R    I    feet       Roll RCS jet Diameter.
C  NPR        R    I    ----      Roll jet nozzle pressure ratio.
C  Q          R    I    lb/sq ft   Roll jet dynamic pressure.
C  S          R    I    sq ft      Wing area
C  T          R    I    lb        RCS roll jet thrust
C  X          R    I    feet       Jet distance ahead of wing trailing
C                                     edge.
C  Y          R    I    feet       Jet distance in from wing tip.
C -----
C RESPIE Common Block
C  NAME      TYPE  I/O  UNITS      DESCRIPTION
C -----
C  RCS_LT(1,500)R  O    ----      Total lift loss from the Reaction
C                                     Control System.
C  V          I    O    ----      Velocity number counter
C FILES USED:
C  LOGICAL UNIT  I/O  DESCRIPTION
C -----
C <list of files used in the routine>
C COMMONS USED:
C  NAME      DESCRIPTION
C -----
C  FIGPIE     conFIGuration for Power Induced Effects - Contains all
C                                     variables needed to define the configuration and other
C                                     parameters of the aircraft.
C  PIEFLAGS   Power Induced Effects FLAGS - Contains variables which
C                                     help keep track of the configuration of the aircraft.
C  PIERCS     Power Induced Effects for the Reaction Control System -
C                                     Contains variables which are sent to the RCSIND
C                                     subroutine.
C  RESPIE     RESults from all POver Induced Effects subroutines -
C                                     Contains results from each subroutine along with

```

```

C          variables for height, velocity, jet deflection angle,
C          and angle of attack and their counters.
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   COPPIE    Controls execution and coordination of Power Induced
C             Effects Module
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   RCSIND    Calculates RCS lift increments while in forward flight
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   05/15/90  KEH -- Completed initial coding and debugging.
C   07/14/91  KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "figpie.inc"
C #include "pieflag.inc"
C #include "piercs.inc"
C #include "respie.inc"
C
C Assign local RCSIND variables to their respective global configuration
C variables.
C   IF (FLGRCS) THEN
C       B = B_W
C       DRCS = D_RCS
C       NPR = PR_RCS
C       Q = Q_RCS
C       S = S_W
C       T = T_RCS
C       X = X_RCS
C       Y = Y_RCS
C       CALL RCSIND
C   ELSE
C       Assign the zero to RCS results.
C       RCS_LT(1,V) = 0.0
C   END IF
C
C   RETURN
C   END

```

Rcsind

```

SUBROUTINE RCSIND
C -----
C ACRONYM: RCS jet INDuced effects.
C   ---
C PURPOSE: Calculation of jet induced lift loss for powered lift
C          aircraft due to jet effects in ground proximity.
C

```

C LOCAL VARIABLES (in addition to the above parameters):

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	KB	R		----	Chordwise position factor.
C	KC	R		----	Spanwise position factor.
C	LTO	R		----	Lift increment for subcritical pressure ratio (does not vary with pressure ratio)
C	LTP	R		----	Lift increment due to pressure ratio
C	LVOLO	R		----	Lift increment for RCS
C	MO	R		lb ft	Rolling moment without losses
C	MV	R		lb ft	Rolling moment with losses
C	PI	R	-	rad	3.1415927
C	VE	R		----	Effective velocity ratio

C GLOBAL VARIABLES (in addition to the above parameters and local vars):

C PIEFLAG Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	RCSFLG	L	O	----	Flag which identifies if RCS lift loss has calculation exceeded the area ratio (Swing / Ajet > 7000)
C					TRUE - Exceeded
C					FALSE - Not Exceeded

C PIERCS Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	B	R	I	feet	Wing span
C	DRCS	R	I	feet	Roll RCS jet Diameter.
C	NPR	R	I	----	Roll jet nozzle pressure ratio.
C	Q	R	I	lb/sq ft	Roll jet dynamic pressure.
C	S	R	I	sq ft	Wing area
C	SAJ	R		----	Wing area / jet area
C	T	R	I	lb	RCS roll jet thrust
C	X	R	I	feet	Jet distance ahead of wing trailing edge.
C	Y	R	I	feet	Jet distance in from wing tip.

C RESPIE Common Block

C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	RCS_LT(1,500)	R	O	----	Total lift loss from the Reaction Control System.
C	V	I	O	----	Velocity number counter
C	VO(500)	R	O	----	Actual velocity value

C FILES USED:

C	LOGICAL UNIT	I/O	DESCRIPTION
---	--------------	-----	-------------

C None.

C COMMONS USED:

C	NAME	DESCRIPTION
C	PIEFLAGS	Power Induced Effects FLAGS - Contains variables which help keep track of the configuration of the aircraft.
C	PIERCS	Power Induced Effects for the Reaction Control System - Contains variables which are sent to the RCSIND

```

C      subroutine.
C      RESPIE      REsults from all Power Induced Effects subroutines -
C                  Contains results from each subroutine along with
C                  variables for height, velocity, jet deflection angle,
C                  and angle of attack and their counters.
C
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   RCSCAL      Isolates and controls execution of RCSIND
C
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   None.
C
C NOTES: None.
C
C REFERENCES:
C   1) None.
C
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C
C NON-STANDARD CODE:
C   ?
C
C AUTHOR(S):
C   Douglas A. Wardwell (DAW), NASA Ames Research Center
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Res Ctr
C
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   02/22/90   DAW -- FORTRAN version of Dick Kuhn's basic program.
C   02/26/90   KEH -- Finished conversion of BASIC program to FORTRAN.
C   07/14/91   KEH -- Eliminated extraneous spacing and fixed comments
C
C-----
C #include "piercs.inc"
C #include "respie.inc"
C #include "pieflag.inc"
C
C *** Local variables.
C      REAL PI, KB, KC, MV, MO, LTO, LTP, VE, LVOL
C
C      PI = 3.1415926
C      MO = T*(B/2.0-Y)
C      VE = SQRT((.00119*(1.69*VO(V))**2.0)/Q)
C      SAJ = (S/2.0)/((PI/4.0)*DRCS**2.0)
C
C
C Calculate chordwise position factor
C      if (Y/DRCS .gt. 10) then
C          KB = 1.0
C      else
C          KB = .25+.2*(Y/DRCS)**.58
C      end if
C
C Calculate spanwise position factor
C      if (X/DRCS .gt. 12.0) then
C          KC = 1.0
C      else
C          KC = .25+.06*(X/DRCS)
C      end if
C
C Limitations of this method
C      IF (VE .GT. .1) THEN
C          LTO = 0.0

```

```

      LTP = 0.0
      go to 15
END IF
C
  IF (SAJ .GT. 7000) THEN
    RCSFLG = .TRUE.
  ELSE
    RCSFLG = .FALSE.
  END IF
C
  if (NPR .lt. 1.893) then
    LTP = 0.0
  else
    LTP = -.017*VE*(SAJ**.42)*(NPR-1.893)**.75
  end if
C
  LTO = ((3.0*VE**3.0)-2.4*VE**2)*SAJ**.5 +
$      (.41*VE**2.2)*SAJ**.688
  LVOLO = (LTO+LTP)*KB*KC
  MV = MO*LVOLO
C Assign the correct value to the overall results.
  RCS_LT(1,V) = LVOLO
100  format (1x,' V, kts ', ' M/Mo ', ' M')
110  format (1x,3(f9.3,3x))
C
15   RETURN
     END

```

Gvcall

SUBROUTINE GVCALL

```

C-----
C ACRONYM:  stolGV CALLing routine for power induced effects
C
C PURPOSE:  The purpose of GVCALL is to set up all correct variables,
C           based on the configuration, that are to be sent to the
C           STOLGV routine. Another reason for this subroutine is all
C           subroutines that need to be called have variables with the
C           same name which need to be kept separate when using common
C           blocks to pass variables back and forth between routines.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C <list of local variables in routine>
C
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C-----
C FIGPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   B_WB      R     I    Ft        Width of Wing-Body
C   D_F       R     I    Ft        Diameter of each Front jet
C   DE_F      R     I    Ft        Effective Diameter of Front jets
C   NUM_F     I     I    ----      NUMBER of Front jets
C   NUM_R     I     I    ----      NUMBER of Rear jets
C   Q_FR      R     I    lb/sq ft   Dynamic pressure for Front and Rear
C   S_B       R     I    Sq Ft      Area of Body
C   S_CS      R     I    Sq Ft      Area of Center Section

```

C	S_F	R	I	Sq Ft	Area of Front jets
C	S_W	R	I	Sq Ft	Area of Wing
C	S_WB	R	I	Sq Ft	Area of Wing-Body
C	TT_F	R	I	----	Front Thrust / total Thrust
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	Z_B	R	I	Ft	Height of body base above nozzle
C	Z_W	R	I	Ft	height of wing above nozzle

C PIEGV Common Block

	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	AFACT	R	I	----	Area ratio - based on whether the wing or body is being used
C	AJ	R	I	Sq Ft	Area of the front jets
C	DE	R	I	Ft	Equivalent Diameter of front jets
C	DF	R	I	Ft	Diameter of front jets
C	DH	R	I	Ft	Height of wing above jets
C	EX	R	I	Ft	Distance from jet pattern center to center of front jets.
C	LTV	R	O	----	Lift loss due to ground vortex.
C	N	I	I	----	Number of front jets
C	Q	R	I	Lb/sq ft	Dynamic pressure of the nozzles
C	TFT	R	I	----	Front thrust split
C	WLB	R	I	----	Width to length ratio of the body.
C	WLJ	R	I	----	Width to length ratio of the jets.

C PIEFLAG Common Block

	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	WBFLAG	L	O	----	Identifies when WingBody planform has been entered by the user. Used to determine when to call MAKEWB subroutine.

C RESPIE Common Block

	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	A	I	O	----	Angle of attack number counter
C	D	I	O	----	Nozzle deflection angle number counter.
C	H	I	O	----	Height number counter
C	LA	R	I	----	Total number of Angle of Attack values
C	LD	R	I	----	Total number of nozzle deflection angles.
C	LH	R	I	----	Total number of height values.
C	LV	R	I	----	Total number of velocity values.
C	REC4	I	O	----	Record number for any file which replaces a 4 X 4 matrix
C	V	I	O	----	Velocity number counter

C FILES USED:

	LOGICAL UNIT	I/O	DESCRIPTION
C	72	O	Direct access file for storage of lift increment due to the ground vortex on the body

```

C      73          O   Direct access file for storage of lift increment
C                      due to the ground vortex on the wing
C
C COMMONS USED:
C   NAME      DESCRIPTION
C   -----
C   FIGPIE    conFIGuration for Power Induced Effects - Contains all
C              variables needed to define the configuration and other
C              parameters of the aircraft.
C   PIEFLAGS  Power Induced Effects FLAGS - Contains variables which
C              help keep track of the configuration of the aircraft.
C   PIEGV     Power Induced Effects for stolGV - Contains
C              variables which are sent to the STOLGV subroutine.
C   RESPIE    RESults from all POver Induced Effects subroutines -
C              Contains results from each subroutine along with
C              variables for height, velocity, jet defelction angle,
C              and angle of attack and their counters.
C
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   COPPIE    Controls execution and coordination of Power Induced
C              Effects Module
C
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   STOLGV    Calculates ground vortex lift increments while in STOL
C              flight
C
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   05/14/90  KEH -- Completed initial coding and debugging.
C   07/13/91  KEH -- Eliminated extraneous spacing and fixed comments
C
C-----
C #include "figpie.inc"
C #include "pieflag.inc"
C #include "piegv.inc"
C #include "respie.inc"
C
C      REC4 = LH * LV * LD * (A - 1) + LH * LV * (D - 1) + LH * (V - 1)
C      $      + H
C Assign local STOLGV variables to their respective global configuration
C variables.
C Variables that do not change with each configuration (Wing, Body, CS)
C      TFT = TT_F
C      WLJ = WL_F
C      AJ = S_F
C      N = NUM_F
C      DE = DE_F
C      DF =D_F
C      Q = Q_FR
C

```

```

IF (NUM_F .EQ. 2 .AND. NUM_R .EQ. 0) THEN
  EX = 0.0
ELSE
  EX = X_FR / 2.0
END IF

C
IF (WBFLAG) THEN
C
  Variables that STOLGV needs while calculating for the Wingbody
C
  IF (B_WB**2 / S_WB .GT. 1.0) THEN
    WLB = 1.0
  ELSE
    WLB = WL_WB
  END IF

  DH = 0.0
  AFACT = (S_WB/S_F)**.62

C
  IF (VO(V) .EQ. 0.0) THEN
    LTV = 0.0
  ELSE
    CALL STOLGV
  END IF

C
  WRITE (72,100,REC=REC4) LTV
  WRITE (73,100,REC=REC4) 0.0
ELSE
C
  Variables that STOLGV needs while calculating for the Body.
  WLB = WL_B
  DH = Z_B
  AFACT = (S_B / S_F * WL_B)**.62

C
  IF (VO(V) .EQ. 0.0) THEN
    LTV = 0.0
  ELSE
    CALL STOLGV
  END IF

C
  WRITE (72,100,REC=REC4) LTV
C
  Variables that STOLGV needs while calculating for the Wing
  WLB = 1.0
  DH = Z_W
  AFACT = ((S_W/S_F)**.62 - (S_CS/S_F)**.62)

C
  IF (VO(V) .EQ. 0.0) THEN
    LTV = 0.0
  ELSE
    CALL STOLGV
  END IF

C
  WRITE (73,100,REC=REC4) LTV
END IF

C
RETURN
C
Format statements
100 FORMAT (F11.5)
END

```


Stolgy

SUBROUTINE STOLGV

C ACRONYM: STOL operation - estimation of Ground Vortex term.

C PURPOSE: The purpose of STOLGV is to calculate the lift loss due to the ground vortex generated from the proximity of the ground.

C LOCAL VARIABLES (in addition to the above parameters):

NAME	TYPE	I/O	UNITS	DESCRIPTION
CP	R	-	----	Ground Vortex factor
H1D	R	-	Ft	Height at which the rate of change of lift with height changes.
H2D	R	-	Ft	Maximum height at which the ground vortex effects are felt
INC	R	I	Deg	Total angle nozzles are facing taking into account the jet deflection angle and the angle of attack.
KX	R	-	----	Factor used in calculating H2D
VE	R	-	----	ratio of the dynamic pressure of the aircraft over the dynamic pressure of the nozzles.

C GLOBAL VARIABLES (in addition to the above local variables):

C PIEGV Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
AFACT	R	I	----	Area ratio - based on whether the wing or body is being used
AJ	R	I	Sq Ft	Area of the front jets
DE	R	I	Ft	Equivalent Diameter of front jets
DF	R	I	Ft	Diameter of front jets
DH	R	I	Ft	Height of wing above jets
EX	R	I	Ft	Distance from jet pattern center to center of front jets.
LTV	R	O	----	Lift loss due to ground vortex.
N	I	I	----	Number of front jets
Q	R	I	Lb/sq ft	Dynamic pressure of the nozzles
TFT	R	I	----	Front thrust split
WLB	R	I	----	Width to length ratio of the body.
WLJ	R	I	----	Width to length ratio of the jets.

C RESPIE Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
A	I	O	----	Angle of attack number counter
D	I	O	----	Nozzle deflection angle number counter.
H	I	O	----	Height number counter
LA	R	I	----	Total number of Angle of Attack values
LD	R	I	----	Total number of nozzle deflection angles.
LH	R	I	----	Total number of height values.
LV	R	I	----	Total number of velocity values.
REC4	I	O	----	Record number for any file which

```

C                                     replaces a 4 X 4 matrix
C                                     Velocity number counter
C   V           I       O   ----
C FILES USED:
C   LOGICAL UNIT   I/O   DESCRIPTION
C   -----
C   <None>
C COMMONS USED:
C   NAME           DESCRIPTION
C   -----
C   PIEFLAGS       Power Induced Effects FLAGS - Contains variables which
C                   help keep track of the configuration of the aircraft.
C   PIEGV          Power Induced Effects for stolGV - Contains
C                   variables which are sent to the STOLGV subroutine.
C   RESPIE         RESults from all Power Induced Effects subroutines -
C                   Contains results from each subroutine along with
C                   variables for height, velocity, jet deflection angle,
C                   and angle of attack and their counters.
C CALLED BY:
C   NAME           DESCRIPTION
C   -----
C   GVCALL         Isolates and controls execution of STOLGV
C ROUTINES CALLED:
C   NAME           DESCRIPTION
C   -----
C   <None>
C NOTES: None.
C REFERENCES:
C   1) Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the
C       Aerodynamic Stability and Control Parameters of STOL Aircraft
C       Configurations." North American Aircraft Operations. Rockwell
C       International Corporation. AFWAL-TR-87-3019. Volume II & III.
C       June 1987.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE           INITIALS & DESCRIPTION
C   05/08/90      KEH -- Initial completion of code.
C   05/14/90      KEH -- Code in working order.
C   07/13/91      KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "piegv.inc"
C #include "respie.inc"
C
C Variable declaration
C   REAL CP, H1D, H2D, INC, KX, VE
C
C Variable initialization
C   INC = DFL(D) + AOA(A)
C   VE=SQRT((0.00119*(1.69*VO(V))**2.)/Q)
C
C   IF (EX/DF .LT. 2.12) THEN
C       KX = 1. + .64 * (1. - (EX/DF)/2.12)
C   ELSE IF (EX/DF .GE. 2.12 .OR.
C           EX/DF .EQ. 0.) THEN
C       $

```

```

      KX = 1.
      END IF
C
      H1D=1.19*((1/VE)**0.68)*(1.+SIN((INC-90.0)/57.296))**0.53-DH/DF
      H2D=(.14/VE**2.0 + 2.0)*SQRT(INC/90.)*WLB**.4 * WLJ**-.12 * KX
      CP=-2.3/SQRT(VE) * (1.0 - HT(H)/(H2D*DF))
      IF (CP .LT. -3.3) CP=-3.3
      IF (CP .GT. 0.0) CP=0.0
C
C Calculate LTV according to its relationship to H1 and H2.
      IF (HT(H) .LT. H1D*DF) THEN
        LTV = TFT * .13 * WLJ**(-.2) * AFACT * VE**.8 * CP *
$         (1.0 + SIN((INC - 90.0)/57.296)) *
$         ((HT(H) + DH)/DF)**-1.3
      ELSE IF (HT(H) .GE. H1D*DF .AND. HT(H) .LE. H2D*DF) THEN
        LTV = TFT * .18 * WLJ**(-.2) * AFACT * VE**(-.5) * CP *
$         (1.0 + SIN((INC - 90.0)/57.296))**2.0 *
$         ((HT(H) + DH)/DF)**-3.2
      ELSE
        LTV = 0.0
      END IF
C
      RETURN
      END

```

Wkcall

```

      SUBROUTINE WKCALL
C-----
C ACRONYM:  stolWK Calling routine for power induced effects
C
C PURPOSE:  The purpose of GVCALL is to set up all the correct
C           variables, based on the configuration, that are to be sent
C           to the STOLWK and JIEPIE routine.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   DE         R    I    ----      Effective diameter of the jets in
C                                     question.
C   LT_WK      R    O    ----      Total Lift loss of the config.
C                                     due to jet wake.
C   LT_WKB     R    O    ----      Lift loss of the body due to jet
C                                     wake.
C   LT_WKW     R    O    ----      Lift loss of the wing due to jet
C                                     wake.
C   LTO        R    -    ----      Lift loss out of ground effect.
C   PER        R    -    ----      Perimeter of jets in question.
C   PI         R    -    ----      Constant - 3.1415926
C
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C-----
C FIGPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   B_B       R    I    Ft          Width of Body
C   B_CS      R    I    Ft          Width of Center Section
C   B_W       R    I    Ft          Width of Wing (Wing span)
C   B_WB      R    I    Ft          Width of Wing-Body

```

C	D_F	R	I	Ft	Diameter of each Front jet
C	D_R	R	I	Ft	Diameter of each Rear jet
C	DE_F	R	I	Ft	Effective Diameter of Front jets
C	DE_R	R	I	Ft	Effective Diameter of Rear jets
C	KLF	R	I	----	Adjustment factor for flap
C					extension when calculating the lift
C					loss due to jet induced effects.
C	MAC	R	I	Ft	Mean Aerodynamic Chord of wing
C	NUM_F	I	I	----	NUMBER of Front jets
C	NUM_R	I	I	----	NUMBER of Rear jets
C	PR_F	R	I	----	Jet Pressure Ratio for front jets
C	PR_FR	R	I	----	Jet Pressure Ratio for all jets
C	PR_R	R	I	----	Jet Pressure Ratio for rear jets
C	Q_F	R	I	lb/sq ft	Dynamic pressure for the front jets
C	Q_R	R	I	lb/sq ft	Dynamic pressure for the rear jets
C	S_B	R	I	Sq Ft	Area of Body
C	S_CS	R	I	Sq Ft	Area of Center Section
C	S_F	R	I	Sq Ft	Area of Front jets
C	S_R	R	I	Sq Ft	Area of Rear jets
C	S_W	R	I	Sq Ft	Area of Wing
C	S_WB	R	I	Sq Ft	Area of Wing-Body
C	SF_FB	R	I	Sq Ft	Area ahead of Front jets using body
C					planform
C	SF_RB	R	I	Sq Ft	Area ahead of Rear jets using the
C					body planform
C	T_F	R	I	lb	Thrust of Front jets
C	T_R	R	I	lb	Thrust of Rear jets
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	XNOZ_F	R	I	Ft	X-coordinates of Front NOzzle
C	XNOZ_R	R	I	Ft	X-coordinates of Rear NOzzle
C	XTE_F	R	I	Ft	X-coordinate of trailing edge
C					for front nozzle (root TE if
C					Nozzle within body)
C	XTE_R	R	I	Ft	X-coordinate of trailing edge
C					for rear nozzle (root TE if
C					Nozzle within body)
C	Y_F	R	I	Ft	Distance between Front jets
C	Y_R	R	I	Ft	Distance between Rear jets
C	YB_F	R	I	Ft	Lateral distance from Body to Front
C					jets (for external jets)
C	YB_R	R	I	Ft	Lateral distance from Body to Rear
C					jets (for external jets)
C	YWB_F	R	I	Ft	Lateral distance from Wingbody to
C					Front jets (for external jets)
C	YWB_R	R	I	Ft	Lateral distance from wingbody to
C					Rear jets (for external jets)
C	Z_B	R	I	Ft	Height of body base above nozzle
C	Z_W	R	I	Ft	height of wing above nozzle
C	-----				
C	PIEFLAG Common Block				
C	NAME	TYPE	I/O	UNITS	DESCRIPTION
C	-----				
C	WBFLAG	L	O	----	Identifies when the WingBody
C					planform has been enter by the
C					user. Used to determine when to

call MAKEWB subroutine.

PIEWK Common Block					DESCRIPTION
NAME	TYPE	I/O	UNITS		
AFACT	R	I	----		Area ratios used in calculating the lift loss due to the jet wake.
AJ	R	I	Sq. Ft		Total area of the jets in question.
AR	R	I	----		Aspect ratio of the planform.
B	R	I	----		Width of current body config.
CU	R	I	----		Momentum coef. based on nozzle.
DH	R	I	Ft		Vertical position of the jet based on the planform.
DI	R	I	Ft		Diameter of the jets in question.
FIGWK	C	-	----		Flag which notifies JIEPIE which planform is in use.
HEIGHT	R	I	Ft		Height of aircraft.
KL_F	R	I	----		Adjustment factor used for flap extension
					1.0 - Flaps extended
					.25 - Flaps not extended
LT	R	O	----		Lift loss returned by STOLWK
LT0	R	-	----		Lift loss out of ground effect.
LT_OG	R	O	----		Out of ground lift loss
MC	R	I	----		Mean Aerodynamic Chord.
N	I	I	----		Number of nozzles based on which nozzle used (Front, Rear)
PR	R	I	----		Pressure Ratio of local jets.
Q	R	I	lb/Sq Ft		Jet dynamic pressure
Q0	R	O	lb/Sq Ft		Free stream dynamic pressure.
S	R	I	Sq Ft		Area of the configuration.
SF	R	I	Sq Ft		Area of configuration that is in front of nozzles.
T	R	I	lb		Thrust of the jets in question.
TEFALG	L	-	----		Flag which notifies STOLWK how close the nozzle is to the trailing edge of the wing.
WL	R	I	----		Width to length ratio of nozzels.
XC	R	-	----		Position of nozzle with respect to wing trailing edge (in % chord)
Y	R	I	Ft		Lateral spacing of jets.
YP	R	I	Ft		Distance of jet center to bodyside.
VEFACT	R	I	----		Effective velocity factor which accounts for the reduction in velocity at the rear nozzles due to the front nozzle.
Z	R	I			Vertical distance of the nozzles.
RESPIE Common Block					DESCRIPTION
NAME	TYPE	I/O	UNITS		
D	I	O	----		Nozzle deflection angle number counter.
H	I	O	----		Height number counter
LD	R	I	----		Total number of nozzle deflection angles.
LH	R	I	----		Total number of height values.
LV	R	I	----		Total number of velocity values.

```

C   REC3      I      O      ----      Record number for any file which
C                                         replaces a 3 X 3 matrix
C   V          I      O      ----      Velocity number counter
C   VO(500)    R      O      ----      Actual velocity value
C FILES USED:
C   LOGICAL UNIT I/O DESCRIPTION
C   -----
C   74          O      Direct access file for storage of lift increment
C                                     due to the jet wake on the body
C   75          O      Direct access file for stroage of lift increment
C                                     due to the jet wake on the wing (with out center
C                                     section)
C COMMONS USED:
C   NAME      DESCRIPTION
C   -----
C   FIGPIE     conFIGuration for Power Induced Effects - Contains all
C                                     variables needed to define the configuration and other
C                                     parameters of the aircraft.
C   PIEFLAGS   Power Induced Effects FLAGS - Contains variables which
C                                     help keep track of the configuration of the aircraft.
C   PIEWK      Power Induced Effects for stolWK and jiepie - Contains
C                                     all variables which are passes to the STOLWK and JIEPIE
C                                     subroutines.
C   RESPIE     RESults from all POWER Induced Effects subroutines -
C                                     Contains results from each subroutine along with
C                                     variables for height, velocity, jet defelction angle,
C                                     and angle of attack and their counters.
C CALLED BY:
C   NAME      DESCRIPTION
C   -----
C   COPPIE     Controls execution and coordination of Power Induced
C                                     Effects Module
C ROUTINES CALLED:
C   NAME      DESCRIPTION
C   -----
C   JIEPIE     Calculates change in lift caused by the jet induce
C                                     effects on a flat plate
C   STOLWK     Calculates jet wake lift increments while in STOL flight
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C AUTHOR(S):
C   Kipp Edward Howard (KEH), Cal Poly San Luis Obispo,
C                                     NASA Ames Moffett Field
C REVISION HISTORY:
C   DATE      INITIALS & DESCRIPTION
C   05/22/90   KEH -- Description.
C   07/14/91   KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "figpie.inc"
C #include "pieflag.inc"
C #include "piewk.inc"
C #include "respie.inc"
C
C Declaration of local variables
C   REAL LT_WK, LT_WKB, LT_WKC, LT_WKW, PI

```

```

C
C Assign local STOLWK variables to their respective global configuration
C variables.
C   Define variables that do not change with planform configuration
C   and nozzle definition.
    HEIGHT = HT(H)
    MC = MAC
    LT_WK = 0.0
    LT_WKB = 0.0
    LT_WKW = 0.0
    KL_F = KLF
    PI = 3.1415926
    Q0 = 0.00119*(1.69*VO(V))**2.0

C
C   IF (WBFLAG) THEN
C       WINGBODY PLANFORM
C       Assign wingbody planform variables.
        AR = B_WB**2.0 / S_WB
        B = B_WB
        S = S_WB
        Z = 0.0
        FIGWK = 'WBDY'
C       Assign variables for the front nozzles.
        AJ = S_F
        DE = DE_F
        DI = D_F
        N = NUM_F
        PER = PI * 2.0 * SQRT(AJ / 3.14159)
        PR = PR_F
        Q = Q_F
        SF = 0.0
        T = T_F
        WL = WL_F
        Y = Y_F
C       YP set to the following value to insure that KL_YP equals 1.0.
        YP = YWB_F
        VEFACT = 1.0
        XC = (XNOZ_F - (XTE_F - MAC))/MAC

C
C       IF (VO(V) .EQ. 0.0) THEN
            LTO = 0.0
            LT_FLP = 0.0
        ELSE
            CU = T / (Q0*VEFACT**2.) / S
C            Calculate Lift loss due to the front nozzle
            CALL JIEPIE
        END IF

C
C       Determine how close the nozzle is to the trailing edge of
C       the wing minus the flap.
C       IF (ABS(XTE_F - XNOZ_F) .LT. MAC/4.0) THEN
            TEFLAG = .TRUE.
        ELSE
            TEFLAG = .FALSE.
        END IF

C
        AFACT = SQRT(S/AJ * B_WB**2.0/S_WB)
        DH = 0.0

```

```

C      LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
      Calculate lift loss due to the jet wake
C      CALL STOLWK
      LT_WKB = LT * TT_F

C      Assign variables for the rear nozzles if necessary
C      IF (NUM_R .GT. 0) THEN
        AJ = S_R
        SF = 0.0
        T = T_R
        DE = DE_R
        DI = D_R
        N = NUM_R
        PER = PI * 2.0 * SQRT(AJ / 3.14159)
        PR = PR_R
        Q = Q_R
        Y = Y_R
        YP = YWB_R
        WL = WL_R
        VEFACT = ((X_FR/D_F) - 1.0)/((X_FR/D_F) + .75)
        XC = (XNOZ_R - (XTE_R - MAC))/MAC

C      IF (VO(V) .EQ. 0.0) THEN
        LTO = 0.0
        LT_FLP = 0.0
      ELSE
        CU = T / (Q0*VEFACT**2.) / S
        CALL JIEPIE
      END IF

C      IF (ABS(XTE_R - XNOZ_R) .LT. MAC/4.0) THEN
        TEFLAG = .TRUE.
      ELSE
        TEFLAG = .FALSE.
      END IF

C      AFACT = SQRT(S/AJ * B_WB**2.0/S_WB)
      DH = 0.0
      LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C      Calculate lift loss due to the jet wake
      CALL STOLWK
      LT_WKB = LT_WKB + LT * TT_R
      LT_WKW = 0.0
    END IF

C      ELSE
C      BODY PLANFORM
C      Assign body planform variables.
      AR = B_B**2.0 / S_B
      S = S_B
      Z = Z_B
      FIGWK = 'BODY'
C      Assign variables for the front nozzles.
      AJ = S_F
      DE = DE_F
      DI = D_F
      N = NUM_F
      PER = PI * 2.0 * SQRT(AJ / 3.14159)

```



```

PR = PR_F
Q = Q_F
SF = SF_FB
T = T_F
WL = WL_F
Y = Y_F

C
C      Use distance to wingbody edge if height of wing is zero,
C      otherwise use the distance to the body edge.
C      IF (Z_W .GT. 0.0) THEN
          YP = YB_F
      ELSE
          YP = YWB_F
      END IF

C
VEFACT = 1.0
XC = 0.0

C
IF (VO(V) .EQ. 0.0) THEN
    LTO = 0.0
    LT_FLP = 0.0
ELSE
    CU = T / (QO*VEFACT**2.) / S
    C      Calculate Lift loss due to the front nozzle
    C      CALL JIEPIE
END IF

C
C      There is no jet flap effect on the body
C      TEFLAG = .FALSE.
AFACT = SQRT(S / AJ * WL_B)
DH = Z_B
LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C      Calculate lift loss due to the jet wake
CALL STOLWK
LT_WKB = LT * TT_F

C
C      Assign variables for the rear nozzles if necessary
C      IF (NUM_R .GT. 0) THEN
          AJ = S_R
          SF = SF_RB
          T = T_R
          DE = DE_R
          DI = D_R
          N = NUM_R
          PER = PI * 2.0 * SQRT(AJ / 3.14159)
          PR = PR_R
          Q = Q_R
          Y = Y_R

C
C      Use distance to wingbody edge if height of wing is zero,
C      otherwise use the distance to the body edge.
C      IF (Z_W .GT. 0.0) THEN
          YP = YB_R
      ELSE
          YP = YWB_R
      END IF

C
WL = WL_R

```

```

XC = 0.0
VEFACT = ((X_FR/D_F) - 1.0)/((X_FR/D_F) + .75)
C
IF (VO(V) .EQ. 0.0) THEN
    LT0 = 0.0
    LT_FLP = 0.0
ELSE
C
    CU = T / (Q0*VEFACT**2.) / S
    CALL JIEPIE
END IF
C
C
    There is no jet flap effect on the body
    TEFLAG = .FALSE.
    AFACT = SQRT(S / AJ * WL_B)
    DH = Z_B
    LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C
    Calculate lift loss due to the jet wake
    CALL STOLWK
    LT_WKB = LT_WKB + LT * TT_R
END IF
C
C
WING PLANFORM
    Assign wing planform variables.
    AR = B_W**2.0 / S_W
    B = B_W
    S = S_W
    Z = Z_W
    FIGWK = 'WING'
C
    Assign variables for the front nozzles.
    AJ = S_F
    DE = DE_F
    DI = D_F
    N = NUM_F
    PER = PI * 2.0 * SQRT(AJ / 3.14159)
    PR = PR_F
    Q = Q_F
    SF = 0.0
    T = T_F
    WL = WL_F
    Y = Y_F
C
    YP set to the following value to insure that KL_YP equals 1.0.
    YP = -1.0 * DI
    VEFACT = 1.0
    XC = (XNOZ_F - (XTE_F - MAC))/MAC
C
IF (VO(V) .EQ. 0.0) THEN
    LT0 = 0.0
    LT_FLP = 0.0
ELSE
C
    CU = T / (Q0*VEFACT**2.) / S
    Calculate Lift loss due to the front nozzle
    CALL JIEPIE
END IF
C
C
    Determine how close the nozzle is to the trailing edge of
    the wing.
    IF (ABS(XTE_F - XNOZ_F) .LT. MAC/4.0) THEN

```

```

      TEFLAG = .TRUE.
ELSE
      TEFLAG = .FALSE.
END IF

C
AFACT = SQRT(S/AJ * B_W**2.0/S_W)-SQRT(S/AJ * B_CS**2.0/S_CS)
DH = Z_W
LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C
Calculate lift loss due to the jet wake
CALL STOLWK
LT_WKW = LT * TT_F

C
Assign variables for the rear nozzles if necessary
C
IF (NUM_R .GT. 0) THEN
  AJ = S_R
  SF = 0.0
  T = T_R
  DE = DE_R
  DI = D_R
  N = NUM_R
  PER = PI * 2.0 * SQRT(AJ / 3.14159)
  PR = PR_R
  Q = Q_R
  Y = Y_R
  YP = -1.0 * DI
  WL = WL_R
  VEFACT = ((X_FR/D_F) - 1.0)/((X_FR/D_F) + .75)
  XC = (XNOZ_R - (XTE_R - MAC))/MAC

C
  IF (VO(V) .EQ. 0.0) THEN
    LTO = 0.0
    LT_FLP = 0.0
  ELSE
    CU = T / (Q0*VEFACT**2.) / S
    CALL JIEPIE
  END IF

C
  IF (ABS(XTE_R - XNOZ_R) .LT. MAC/4.0) THEN
    TEFLAG = .TRUE.
  ELSE
    TEFLAG = .FALSE.
  END IF

C
  AFACT = SQRT(S/AJ * B_W**2.0/S_W)-SQRT(S/AJ *
$      B_CS**2.0/S_CS)
  DH = Z_W
  LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C
  Calculate lift loss due to the jet wake
  CALL STOLWK
  LT_WKW = LT_WKW + LT * TT_R
END IF

C
C
C
CENTER SECTION PLANFORM
Assign center section planform variables.
AR = B_CS**2.0 / S_CS
B = B_CS
S = S_CS
Z = Z_W

```

```

FIGWK = 'CSEC'
C Assign variables for the front nozzles.
AJ = S_F
DE = DE_F
DI = D_F
N = NUM_F
PER = PI * 2.0 * SQRT(AJ / 3.14159)
PR = PR_F
Q = Q_F
SF = 0.0
T = T_F
WL = WL_F
Y = Y_F
C YP set to the following value to insure that KL_YP equals 1.0.
YP = -1.0 * DI
VEFACT = 1.0
XC = 0.0

C
IF (VO(V) .EQ. 0.0) THEN
    LT0 = 0.0
    LT_FLP = 0.0
ELSE
    CU = T / (Q0*VEFACT**2.) / S
    C Calculate Lift loss due to the front nozzle
    CALL JIEPIE
END IF

C
TEFLAG = .FALSE.
AFACT = SQRT(S/AJ * B_W**2.0/S_W)-SQRT(S/AJ * B_CS**2.0/S_CS)
DH = Z_W
LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C Calculate lift loss due to the jet wake
CALL STOLWK
LT_WKC = LT * TT_F

C
C Assign variables for the rear nozzles if necessary
IF (NUM_R .GT. 0) THEN
    AJ = S_R
    DE = DE_R
    DI = D_R
    N = NUM_R
    PER = PI * 2.0 * SQRT(AJ / 3.14159)
    PR = PR_R
    Q = Q_R
    SF = 0.0
    T = T_R
    Y = Y_R
    YP = -1.0 * DI
    WL = WL_R
    VEFACT = ((X_FR/D_F) - 1.0)/((X_FR/D_F) + .75)
    XC = 0.0

C
IF (VO(V) .EQ. 0.0) THEN
    LT0 = 0.0
    LT_FLP = 0.0
ELSE
    CU = T / (Q0*VEFACT**2.) / S
    CALL JIEPIE

```

```

      END IF
C
      TEFLAG = .FALSE.
      AFACT = SQRT(S/AJ * B_W**2.0/S_W)-SQRT(S/AJ *
$      B_CS**2.0/S_CS)
      DH = Z_W
      LT_OG = -0.00022*SQRT(S/AJ)*(PR_FR**-0.64)*(PER/DE)**1.58
C      Calculate lift loss due to the jet wake
      CALL STOLWK
      LT_WKC = LT_WKC + LT * TT_R
      END IF
C
      END IF
C
      REC3 = LH * LV * (D - 1) + LH * (V - 1) + H
C      Lift loss due to wake for body
      WRITE (74,100,REC=REC3) LT_WKB
C      Lift loss due to wake for wing
      WRITE (75,100,REC=REC3) LT_WKW - LT_WKC
C
      RETURN
C
C      Format Statements
100  FORMAT (F11.5)
      END

```

Jiepie

SUBROUTINE JIEPIE

```

C-----
C ACRONYM:  Jet Induce Effects for the Power Induced Effects module.
C
C PURPOSE:  The purpose of JIEPIE is to calculate the change in the lift
C           coefficient caused by the jet induced effects on a flat
C           plate. JIEPIE takes in to account aspect ratio, longitudinal
C           vertical, and lateral position of jets, defelction angle,
C           distance from side of body, and non-circular or closely
C           spaced jets..
C LOCAL VARIABLES (in addition to the above parameters):
C
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   CL_BAS      R    -    ----      Basic lift loss on a square
C                                     planform with the jet at the center
C                                     of the planform.
C   FP_BAS      R    -    ----      Basic lift gain due to jet flap
C                                     action
C   K_JFH       R    -    ----      Adjustment factor Jet Flap
C                                     calculation for aircraft heigt.
C   K_PR        R    -    ----      Adjustment factor for Net Jet
C                                     Pressure Ratio
C   KL_A        R    -    ----      Adjustment factor for aspect ratio
C   KL_B        R    -    ----      Adjust for the "jet-flap" span
C   KL_D        R    -    ----      Adjustment factor for the jet
C                                     deflection angle.
C   KL_N        R    -    ----      Adjustment factor for non-circular
C                                     nozzle configuration(nad closely
C                                     spaced jets).
C   KL_X1       R    -    ----      Adjustment factor for longitudinal

```

position of jet (jet interference term)

Adjustment factor for longitudinal position of jet (jet flap term)

Adjustment factor for lateral spacing of the jets

Adjustment factor for distance of jet center from side of the body.

Adjustment factor for the vertical position of the jet.

Effective velocity ratio

GLOBAL VARIABLES (in addition to the above parameters and local vars):

PIEWK Common Block

NAME	TYPE	I/O	UNITS	DESCRIPTION
AFACT	R	I	----	Area ratios used in calculating the lift loss due to the jet wake.
AJ	R	I	Sq. Ft	Total area of the jets in question.
AR	R	I	----	Aspect ratio of the planform.
B	R	I	----	Width of current body config.
CU	R	I	----	Momentum coef. based on nozzle.
DH	R	I	Ft	Vertical position of the jet based on the planform.
DI	R	I	Ft	Diameter of the jets in question.
FIGWK	C	-	----	Flag which notifies JIEPIE which planform is in use.
HEIGHT	R	I	Ft	Height of aircraft.
KL_F	R	I	----	Adjustment factor used for flap extension 1.0 - Flaps extended .25 - Flaps not extended
LT	R	O	----	Lift loss returned by STOLWK
LT_FLP	R	O	----	Basic lift gain due to jet flap action.
LT0	R	-	----	Lift loss out of ground effect.
LT_0G	R	O	----	Out of ground lift loss
MC	R	I	----	Mean Aerodynamic Chord.
N	I	I	----	Number of nozzles based on which nozzle used (Front, Rear)
PR	R	I	----	Pressure Ratio of local jets.
Q	R	I	lb/Sq Ft	Jet dynamic pressure
Q0	R	O	lb/Sq Ft	Free stream dynamic pressure.
S	R	I	Sq Ft	Area of the configuration.
SF	R	I	Sq Ft	Area of configuration that is in front of nozzles.
T	R	I	lb	Thrust of the jets in question.
TEFALG	L	-	----	Flag which notifies STOLWK how close the nozzle is to the trailing edge of the wing.
WL	R	I	----	Width to length ratio of nozzles.
XC	R	-	----	Position of nozzle with respect to wing trailing edge (in % chord)
Y	R	I	Ft	Lateral spacing of jets.
YP	R	I	Ft	Distance of jet center to bodyside.
VEFACT	R	I	----	Effective velocity factor which accounts for the reduction in velocity at the rear nozzles due to

the front nozzle.
Vertical distance of the nozzles.

NAME	TYPE	I/O	UNITS	DESCRIPTION
D	I	O	----	Nozzle deflection angle number counter.
H	I	O	----	Height number counter
LD	R	I	----	Total number of nozzle deflection angles.
LH	R	I	----	Total number of height values.
LV	R	I	----	Total number of velocity values.
REC3	I	O	----	Record number for any file which replaces a 3 X 3 matrix
V	I	O	----	Velocity number counter
VO(500)	R	O	----	Actual velocity value

FILES USED:

LOGICAL UNIT	I/O	DESCRIPTION
<None>		

COMMONS USED:

NAME	DESCRIPTION
PIEWK	Power Induced Effects for stolWK and jiepie - Contains all variables which are passes to the STOLWK and JIEPIE subroutines.
RESPIE	RESults from all PIE subroutines - Contains variables which are the main outputs from each subroutine

CALLED BY:

NAME	DESCRIPTION
WKCALL	Isolates and controls execution of STOLWK

ROUTINES CALLED:

NAME	DESCRIPTION
<None>	

NOTES: None.

REFERENCES:

- 1) Henderson, C., Clark, J., Walters, M. "V/STOL Aerodynamics and Stability & Control Manual" Naval Air Development Center, Pennsylvania, January 15, 1980. NADC-80017-60.
- 2) Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the Aerodynamic Stability and Control Parameters of STOL Aircraft Configurations." North American Aircraft Operations. Rockwell International Corporation. AFWAL-TR-87-3019. Volume II & III. June 1987.

ENVIRONMENT:

FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.

NON-STANDARD CODE:

?

AUTHOR(S):

Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett

REVISION HISTORY:

DATE	INITIALS & DESCRIPTION
05/08/90	KEH -- Initial completion of code.
07/13/91	KEH -- Eliminated extraneous spacing and fixed comments

```

#include "piewk.inc"
#include "respie.inc"
C
C Variable declaration
  REAL CL_BAS, FP_BAS, VE, KL_A, KL_B, KL_X1, kl_X2, KL_Y, KL_Z,
  $    KL_YP, KL_D, K_JFH, KL_N, K_PR
C
C Determine the effective velocity ratio
  VE=VEFACT * SQRT(Q0/Q)
C Calculate Cl,basic (lift loss on a square planform with a jet at the
C center of the planform. (AFWAL-TR-87-3019 vol III, pg E-4)
  CL_BAS = -3.0 * VE**2.0 * (SQRT(S/AJ) - 1.0)**(2.0/3.0) +
  $    35.0 * VE**5.5 * (SQRT(S/AJ) - 1.0)
C Calculate Basic lift gain due to jet flap action (AFWAL-TR-87-3019
C Vol II Sec. 4.1.3.1 pg 4-13)
  FP_BAS = ((AR + 2.0 * CU / 3.1415926) / (AR + 2.0 + .604 *
  $    SQRT(CU) + .876 * CU) * 6.3 * CU**.63 - CU *
  $    SIN(DFL(D)/57.296)) / CU
C
  IF (FP_BAS * CU .GT. 1.94 * AR) THEN
    FP_BAS = 1.94 * AR / CU
  ELSE IF (FP_BAS * CU .GT. 12.57) THEN
    FP_BAS = 12.57 / CU
  ELSE IF (FP_BAS .LT. 0.0) THEN
    FP_BAS = 0.0
  END IF
C
C Adjust for Aspect Ratio
  IF (AR .LT. 1.0) THEN
    KL_A = AR**.05 - .5 * (1.0 - AR)**3.6
  ELSE
    KL_A = 1.0 - .4 * (1.0 - 1.0 / AR)**4.0
  END IF
C
C Adjust for longitudinal position of the jet
C Bodies
  IF (AR .LT. 1.0) THEN
    KL_X1 = 1.35 - .7 * (SF/S) - 2.5 * (ABS(SF/S - 0.5))**2.5
    KL_X2 = 0.0
C Wings
  ELSE IF (AR .GE. 1.0) THEN
C
    IF (XC .LE. .75) THEN
      KL_X1 = 1.0
      KL_X2 = 0.0
    ELSE IF (XC .GT. .75 .AND. XC .LE. 1.5) THEN
      KL_X1 = 0.0
      KL_X2 = 1.0
    ELSE IF (XC .GT. 1.5 .AND. XC .LT. 3.5) THEN
      KL_X1 = 0.0
      KL_X2 = -.5 * XC + 1.75
    ELSE IF (XC .GT. 3.5) THEN
      KL_X1 = 0.0
      KL_X2 = 0.0
    END IF
C
  END IF
C

```



```

      IF (FIGWK .EQ. 'CSEC') THEN
        KL_X1 = 1.0
        KL_X2 = 0.0
      END IF
C
C Adjust for the vertical position of the jet.
      KL_Z = 1.0 - .15 * ABS(Z/DI)
C
C Adjust for lateral spacing of the jets
      IF (Y/DI .LT. 3.0 .AND. Y .NE. 0.0) THEN
        KL_Y = 1.2 - .1 * (Y/DI - 1.0)
      ELSE
        KL_Y = 1.0
      END IF
C
C Adjust for the distance of the jet center from the side of the body.
      IF (YP/DI .LT. -.55) THEN
        KL_YP = 1.0
      ELSE IF (YP/DI .LT. 0.0) THEN
        KL_YP = .85 - .273 * YP/DI
      ELSE IF (YP/DI .LT. 2.0) THEN
        KL_YP = .85 - .425 * YP/DI
      ELSE
        KL_YP = 0.0
      END IF
C
C Adjust for the jet deflection angle.
      KL_D = DFL(D) / 90.0
C Adjust for non-circular nozzle configuration(nad closely spaced jets).
      KL_N = 1.0 - (14.0 * VE**1.5 - 22.0 * VE**2.5) * (1.0 - WL**3)
C
C Adjust for the "jet-flap" span
      IF (FIGWK .EQ. 'WING' .OR. AR .GE. 1.0) THEN
        KL_B = N * DI / B
      ELSE
        KL_B = 0.0
      END IF
C
C Adjust for Net Jet Pressure Ratio
      IF (PR .GE. 1.893) THEN
        K_PR = (PR/1.893)**.25
      ELSE
        K_PR = 1.0
      END IF
C
C Adjust Jet Flap calculation for aircraft height.
      K_JFH = 1.0 - .055 * (MC/HEIGHT)**1.3
C*****
C Make final calculation for Cl
C Lift loss due to jet interference
      LT0 = CL_BAS * KL_A * KL_X1 * KL_Y * KL_Z * KL_YP * KL_D * KL_N
      $      * K_PR
C Lift gain due to jet flap interaction
      LT_FLP = FP_BAS * KL_X2 * KL_Z * KL_D * KL_B * KL_F * K_JFH
C
      RETURN
      END

```

Stolwk

SUBROUTINE STOLWK

```

C-----
C ACRONYM:  STOL operation - estimation of jet Wake term
C
C PURPOSE:  The purpose of STOLWK is to calculate the lift loss due
C           to the jet wake system.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   VE         R    -    ----      Effective velocity ratio
C   KX         R    -    ----      Factor used in calculating H2D
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C PIEWK Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   AFACT      R    I    ----      Area ratios used in calculating
C                                     the lift loss due to the jet wake.
C   AR         R    I    ----      Aspect ratio of the planform.
C   DH         R    I    Ft        Vertical position of the jet based
C                                     on the planform.
C   DI         R    I    Ft        Diameter of the jets in question.
C   LT         R    O    ----      Lift loss returned by STOLWK
C   LT_FLP     R    O    ----      Basic lift gain due to jet flap
C                                     action.
C   LTO        R    -    ----      Lift loss out of ground effect.
C   Q          R    I    lb/Sq Ft  Jet dynamic pressure
C   Q0         R    O    lb/Sq Ft  Free stream dynamic pressure.
C   TEFALG     L    -    ----      Flag which notifies STOLWK how
C                                     close the nozzle is to the trailing
C                                     edge of the wing.
C   WL         R    I    ----      Width to length ratio of nozzels.
C   VEFACT     R    I    ----      Effective velocity factor which
C                                     accounts for the reduction in
C                                     velocity at the rear nozzles due to
C                                     the front nozzle.
C-----
C RESPIE Common Block
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   D          I    O    ----      Nozzle deflection angle number
C                                     counter.
C   DFL(500)   R    I    deg       Actual nozzle deflection angle
C                                     value
C   H          I    O    ----      Height number counter
C   HT(500)    R    O    Ft        Actual height value
C FILES USED:
C   LOGICAL UNIT  I/O  DESCRIPTION
C   -----
C   (none)
C COMMONS USED:
C   NAME          DESCRIPTION
C   -----
C   PIEWK         Power Induced Effects for stolWK and jiepie - Contains
C                 all variables which are passes to the STOLWK and JIEPIE
C                 subroutines.

```

```

C   RESPIE      RESults from all PIE subroutines - Contains variables
C               which are the main outputs from each subroutine
C CALLED BY:
C   NAME        DESCRIPTION
C   -----
C   WKCALL      Isolates and controls execution of STOLWK
C ROUTINES CALLED:
C   NAME        DESCRIPTION
C   -----
C   <None>
C NOTES: None.
C REFERENCES:
C   1) Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the
C      Aerodynamic Stability and Control Parameters of STOL Aircraft
C      Configurations." North American Aircraft Operations. Rockwell
C      International Corporation. AFWAL-TR-87-3019. Volume II & III.
C      June 1987.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   ?
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE        INITIALS & DESCRIPTION
C   05/09/90    KEH -- Initial completion of code.
C   07/13/91    KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "respie.inc"
C #include "piewk.inc"
C
C   REAL VE, KX
C
C Determine the effective velocity ratio
C   VE=VEFACT * SQRT(Q0/Q)
C Set KX equal to 1 because KX is based on the distance between the
C center of the jet pattern to the center of the front jet(s). The
C lift loss due to the jet wake is calculated for the front jets and
C rear jets individually, so the distance from the center of the jet
C pattern to the center of the front jet(s) is always going to be zero
C which causes KX to equal 1.0.
C   KX = 1.0
C
C Test to see which equation to use (near or far away from the TE)
C   IF (AR .GE. 1.0 .AND. TEFLAG) THEN
C       Near the trailing edge of the wing (Lift gain)
C
C       If VE equals zero then set LT to LT_FLP
C       IF (VE .EQ. 0.0) THEN
C           LT = LT_FLP
C       ELSE
C           LT = LT_FLP * (1.0 + .01/VE**2.0 * AFACT
C               * (DFL(D)/90.0)**1.34 * ((HT(H) + DH)/DI)**-1.27)
C   $
C   END IF
C
C ELSE
C   Not near the trailing edge of the wing (Lift loss)
C   LT = LT0 * (1.0 - .7 * AFACT * (DFL(D)/90.0)**1.34

```


3) Output necessary data				
C				Boundary layer factor
C	KB	R	I	----
C	KR	R	I	----
C	L_B	R	I	Ft
C	L_CS	R	I	Ft
C	L_F	R	I	Ft
C	L_R	R	I	Ft
C	L_WB	R	I	Ft
C	MAC	R	I	Ft
C	MD_RCS	R	I	lbm/s
C	N_BODY	I	I	----
C				Number of data points for Body planform
C	N_WING	I	I	----
C				Number of data points for Wing planform
C	N_WB	I	I	----
C				Number of data points for Wing-Body planform
C	NUM	I	I	Ft
C	NUM_F	I	I	----
C	NUM_R	I	I	----
C	PER_FR	R	I	Ft
C	PP_LID	R	I	----
C				Total perimeter of jets
C	PR_FR	R	I	----
C	PR_RCS	R	I	----
C	PT_RCS	R	I	lb/sq ft
C	Q_FR	R	I	lb/sq ft
C				Ratio of perimeter enclosed by lids to total perimeterC
C	Q_RCS	R	I	lb/sq ft
C	S_B	R	I	Sq Ft
C	S_CS	R	I	Sq Ft
C	S_JP	R	I	Sq Ft
C	S_F	R	I	Sq Ft
C	S_FR	R	I	Sq Ft
C	S_LID	R	I	Sq Ft
C	S_R	R	I	Sq Ft
C	S_W	R	I	Sq Ft
C	S_WB	R	I	Sq Ft
C	SF_FB	R	I	Sq Ft
C				Jet Pressure Ratio for all jets
C	SF_FWB	R	I	Sq Ft
C				Roll RCS Pressure Ratio
C	SF_RB	R	I	Sq Ft
C	SF_RWB	R	I	Sq Ft
C				Total pressure for one roll RCS jet
C	SP_JP	R	I	Sq Ft
C				Dynamic pressure for Front and Rear jets
C	SPLY_F	R	I	Ft
C	SPLY_R	R	I	Ft
C	T_F	R	I	lb
C	T_R	R	I	lb
C	T_RCS	R	I	lb
C	TP_RCS	R	I	Rankin
C				Dynamic pressure for roll RCS jets
C	TT_F	R	I	----
C	TT_R	R	I	----
C	VEND	R	I	kts
C	VO	R	I	kts
C	VSTART	R	I	kts
				Area of Body
				Area of Center Section
				Area enclosed by Jet Pattern
				Area of Front jets
				Total jet exit area
				Area enclosed by LIDS
				Area of Rear jets
				Area of Wing
				Area of Wing-Body
				Area ahead of Front jets using body planform
				Area ahead of Front jets using the wingbody planform
				Area ahead of Rear jets
				Area ahead of Rear jets using the wingbody planform
				Actual surface area within area enclosed by nozzles
				SPLaY angle of Front jet
				SPLaY angle of Rear jet
				Thrust of Front jets
				Thrust of Rear jets
				Thrust of roll RCS nozzle
				Temperature of flow in roll RCS nozzle
				Front Thrust / total Thrust
				Rear Thrust / total Thrust
				Ending value for aircraft velocity
				Forward velocity of aircraft
				Starting value for aircraft

C					velocity
C	VSTEP	R	I	kts	Step value for aircraft velocity
C	W_F	R	I	Ft	Width of Front jet
C	W_R	R	I	Ft	Width of Rear jets
C	WIWE	R	I	----	Flow Weight of Inlet / Flow Weight of Exit
C	WL_B	R	I	----	Width to Length ratio of Body
C	WL_CS	R	I	----	Width to Length ratio of Center Section
C	WL_JP	R	I	----	Width to Length ratio of Jet Pattern
C	WL_F	R	I	----	Width to Length ratio of Front jets
C	WL_R	R	I	----	Width to Length ratio of Rear jets
C	WL_WB	R	I	----	Width to Length ratio of Wing-Body
C	X_BODY(500)	R	I	Ft	X-coordinates of Body planform
C	X_FR	R	I	Ft	Distance between Front & Rear jets
C	X_RCS	R	I	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
C	X_WB(500)	R	I	Ft	X-coordinates of Wing-Body planform
C	X_WING(500)	R	I	Ft	X-coordinates of Wing planform
C	XCA.CG	R	I	Ft	Distance of center of area ahead of CG
C	X_CA	R	I	Ft	X-coordinate of Center of Area
C	X.CG	R	I	Ft	X-coordinate of Center of Gravity
C	XCG_C2	R	I	Ft	Distance from CG to MAC/2
C	XCG_C4	R	I	Ft	Distance from CG to MAC/4
C	XCG_F	R	I	Ft	Distance Front jet is ahead of CG
C	XCG_I	R	I	Ft	Inlet longitudinal distance ahead of CG
C	XCG_R	R	I	Ft	Distance Rear jet is ahead of CG
C	XNOZ_F	R	I	Ft	X-coordinates of Front NOzzle
C	XNOZ_R	R	I	Ft	X-coordinates of Rear NOzzle
C	Y_BODY(500)	R	I	Ft	Y-coordinates of Body planform
C	Y_F	R	I	Ft	Distance between Front jets
C	YNOZ_F	R	I	Ft	Y-coordinates of Front NOzzle
C	YNOZ_R	R	I	Ft	Y-coordinates of R NOzzle
C	Y_R	R	I	Ft	Distance between Rear jets
C	Y_RCS	R	I	Ft	Distance of roll RCS nozzle in from wingtip
C	Y_WB(500)	R	I	Ft	Y-coordinates of Wing-Body planform
C	Y_WING(500)	R	I	Ft	Y-coordinates of Wing planform
C	YB_F	R	I	Ft	Lateral distance from Body to Front jets (for external jets)
C	YB_R	R	I	Ft	Lateral distance from Body to Rear jets (for external jets)
C	Z_B	R	I	Ft	Height of body base above nozzle
C	ZCG_I	R	I	Ft	Inlet vertical distance above CG
C	Z_W	R	I	Ft	height of wing above nozzle

C FILES USED:

C	LOGICAL UNIT	I/O	DESCRIPTION
C	68	O	Sequential file for table output
C	70	O	Direct access file for storage of lift increment due to suckdown
C	71	O	Direct access file for storage of lift increment due to fountain
C	72	O	Direct access file for storage of lift increment due to the ground vortex on the body

```

C 73      O Direct access file for storage of lift increment
C          due to the ground vortex on the wing
C 74      O Direct access file for storage of lift increment
C          due to the jet wake on the body
C 75      O Direct access file for stroage of lift increment
C          due to the jet wake on the wing (with out center
C          section)

```

COMMONS USED:

NAME	DESCRIPTION
FIGPIE	conFIGuration for Power Induced Effects - Contains all variables needed to define the configuration and other parameters of the aircraft.
PIEFLAGS	Power Induced Effects FLAGS - Contains variables which help keep track of the configuration of the aircraft.
PIERROR	Power Induced Effects for eRRORS - Contains all error flags within PIE.
RESPIE	RESults from all Power Induced Effects subroutines - Contains results from each subroutine along with variables for height, velocity, jet defelction angle, and angle of attack and their counters.

CALLED BY:

NAME	DESCRIPTION
COPPIE	Controls execution and coordination of Power Induced Effects Module

ROUTINES CALLED:

NAME	DESCRIPTION
JIETAB	Creates output file which is used by ACSYNT

C NOTES: None.

C REFERENCES:

C 1) None.

C ENVIRONMENT:

C FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.

C NON-STANDARD CODE:

C ?.

C AUTHOR(S):

C Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Res Ctr.

C REVISION HISTORY:

DATE	INITIALS & DESCRIPTION
04/03/90	KEH -- Initial coding complete.
07/14/91	KEH -- Eliminated extraneous spacing and fixed comments

```

C
C #include "figpie.inc"
C #include "pieflag.inc"
C #include "pierror.inc"
C #include "respie.inc"

```

```

C
C REAL PI, DIV
C INTEGER CHOICE

```

```

C
C PI = 3.1415926

```

C Print out all input variables

```

C WRITE (6,997) CONFIG, B_B, B_W, B_WB, B_JP, B_CS, L_B, L_WB,
C $ S_JP, L_CS, S_B, S_W, S_WB, S_LID, S_CS, DB_B, DB_W, DB_WB,
C $ SP_JP, DB_CS, WL_B, MAC, WL_WB, WL_JP, WL_CS, Z_B, Z_W, PP_LID, KR

```

```

WRITE (6,998)
$ NUM_F, NUM_R, NUM, PR_FR, D_F, D_R, DR, DE_F, DE_R, DE_FR, WIWE,
$ S_F, S_R, S_FR, XCG_I, SPLY_F, SPLY_R, PER_FR, ZCG_I,
$ W_F, W_R, Q_FR, L_F, L_R, X_FR, WL_F, WL_R,
$ Y_F, Y_R, YB_F, YB_R, YWB_F, YWB_R, PR_F, PR_R, T_F, T_R,
$ TT_F, TT_R, Q_F, Q_R, XTE_F, XTE_R,
$ XCG_F, XCG_R, SF_FB, SF_RB, SF_FWB, SF_RWB, XNOZ_F, XNOZ_R, YNOZ_F, YNOZ_R
WRITE(6,999)
$ E_1, D_RCS, XCG_C2, E_3, PR_RCS, XCG_C4, E_4, T_RCS, XCA.CG,
$ THA_1*180./PI, TP_RCS, KB, THA_3*180./PI, PT_RCS, KLF,
$ THA_4*180./PI, MD_RCS, X.CG, S_FA1, Q_RCS, X_CA, S_FA3, X_RCS,
$ S_FA4, Y_RCS, SP_FA1, SP_FA3, SP_FA4, Y_1, Y_3, Y_4,
$ YP_1, YP_3, YP_4, SCALE3

C
  IF (N_BODY .GE. N_WING .AND. N_WB .EQ. 0) THEN
    N = N_BODY
  ELSE IF (N_BODY .LT. N_WING .AND. N_WB .EQ. 0) THEN
    N = N_WING
  ELSE
    N = N_WB
  END IF

C
  WRITE (6,990)
  DO I = 1,N

C
    IF (I .LE. N_CS .AND. I .LE. N_WING) THEN
      WRITE (6,995) X_BODY(I), Y_BODY(I), X_WING(I), Y_WING(I),
        X_WB(I), Y_WB(I), X_CS(I), Y_CS(I)
    $
    ELSE IF (I .LE. N_BODY .AND. I .LE. N_WING) THEN
      WRITE (6,991) X_BODY(I), Y_BODY(I), X_WING(I), Y_WING(I),
        X_WB(I), Y_WB(I)
    $
    ELSE IF (I .GT. N_BODY .AND. I .LE. N_WING) THEN
      WRITE (6,992) X_WING(I), Y_WING(I), X_WB(I), Y_WB(I)
    ELSE IF (I .LE. N_BODY .AND. I .GT. N_WING .AND.
    $
      I .GT. N_CS) THEN
      WRITE (6,993) X_BODY(I), Y_BODY(I), X_WB(I), Y_WB(I)
    ELSE IF (I .LE. N_CS .AND. I .GT. N_WING) THEN
      WRITE (6,996) X_BODY(I), Y_BODY(I), X_WB(I), Y_WB(I),
    $
      X_CS(I), Y_CS(I)
    ELSE
      WRITE (6,994) X_WB(I), Y_WB(I)
    END IF

C
  END DO
  WRITE (6,800) DBERR, SDAERR, T_ERR
990  FORMAT (/7X,'Body',13X,'Wing',11X,'Wingbody',8X,'Center Sec',/5X,
  $ 'X',6X,'Y',3(9X,'X',6X,'Y'))
991  FORMAT (1X,2(F7.2),2(3X,2(F7.2)))
992  FORMAT (1X,16X,2(F7.2),3X,2(F7.2))
993  FORMAT (1X,2(F7.2),20X,2(F7.2))
994  FORMAT (35X,2(F7.2))
995  FORMAT (1X,2(F7.2),3(3X,2(F7.2)))
996  FORMAT (1X,2(F7.2),17X,2(3X,2(F7.2)))
997  FORMAT (1X,A18/
  $ 1X,'Body',11X,'Wing',11X,'Wing-Body',6X,'Jet Pattern ',
  $ 1X,'Center Section',
  $ /1X,'-----',1X,'-----',1X,'-----',
  $ 1X,'-----',1X,'-----',

```



```

$ /' B_B='F7.2,' B_W='F7.2,' B_WB='F7.2,' B_JP='F7.2,
$ /' B_CS='F7.2,
$ /' L_B='F7.2,15X,' L_WB='F7.2,' S_JP='F7.2,' L_CS=',
$ F7.2,
$ /' S_B='F7.2,' S_W='F7.2,' S_WB='F7.2,' S_LID='F7.2,
$ /' S_CS='F7.2,
$ /' DB_B='F7.2,' DB_W='F7.2,' DB_WB='F7.2,' SP_JP='F7.2,
$ /' DB_CS='F7.2,
$ /' WL_B='F7.2,' MAC='F7.2,' WL_WB='F7.2,' WL_JP='F7.2,
$ /' WL_CS='F7.2,
$ /' Z_B='F7.2,' Z_W='F7.2,16X,' PP_LID='F7.2,
$ /' KR='F7.2)
998 FORMAT(
$ /1X,'Front Nozzles',2X,'Rear Nozzles',3X,'Both Nozzles',3X,
$ 'Misc. Nozzles',
$ /1X,'-----',1X,'-----',1X,'-----',
$ 1X,'-----',
$ /' NUM_F='I7,' NUM_R='I7,' NUM='I7,' PR_FR='F7.2,
$ /' D_F='F7.2,' D_R='F7.2,15X,' DR='F7.2,
$ /' DE_F='F7.2,' DE_R='F7.2,' DE_FR='F7.2,' WIWE='F7.2,
$ /' S_F='F7.2,' S_R='F7.2,' S_FR='F7.2,' XCG_I='F7.2,
$ /' SPLY_F='F7.2,' SPLY_R='F7.2,' PER_FR='F7.2,' ZCG_I='F7.2,
$ /' W_F='F7.2,' W_R='F7.2,' Q_FR='F7.1,
$ /' L_F='F7.2,' L_R='F7.2,' X_FR='F7.2,
$ /' WL_F='F7.2,' WL_R='F7.2,
$ /' Y_F='F7.2,' Y_R='F7.2,
$ /' YB_F='F7.2,' YB_R='F7.2,
$ /' YWB_F='F7.2,' YWB_R='F7.2,
$ /' PR_F='F7.2,' PR_R='F7.2,
$ /' T_F='F7.1,' T_R='F7.1,
$ /' TT_F='F7.2,' TT_R='F7.2,
$ /' Q_F='F7.1,' Q_R='F7.1,
$ /' XTE_F='F7.2,' XTE_R='F7.2,
$ /' XCG_F='F7.2,' XCG_R='F7.2,
$ /' SF_FB='F7.2,' SF_RB='F7.2,
$ /' SF_FWB='F7.2,' SF_RWB='F7.2,
$ /' XNOZ_F='F7.2,' XNOZ_R='F7.2,
$ /' YNOZ_F='F7.2,' YNOZ_R='F7.2)
999 FORMAT(
$ /1X,'Fountain Arms',2X,'RCS Nozzles',4X,'Misc.',
$ /1X,'-----',1X,'-----',1X,'-----',
$ /' E_1='F7.2,' D_RCS='F7.2,' XCG_C2='F7.2,
$ /' E_3='F7.2,' PR_RCS='F7.2,' XCG_C4='F7.2,
$ /' E_4='F7.2,' T_RCS='F7.2,' XCA_CG='F7.2,
$ /' THA_1='F7.2,' TP_RCS='F7.2,' KB='F7.2,
$ /' THA_3='F7.2,' PT_RCS='F7.2,' KLF='F7.2,
$ /' THA_4='F7.2,' MD_RCS='F7.2,' X_CG='F7.2,
$ /' S_FA1='F7.2,' Q_RCS='F7.1,' X_CA='F7.2,
$ /' S_FA3='F7.2,' X_RCS='F7.2,
$ /' S_FA4='F7.2,' Y_RCS='F7.2,
$ /' SP_FA1='F7.2,
$ /' SP_FA3='F7.2,
$ /' SP_FA4='F7.2,
$ /' Y_1='F7.2,
$ /' Y_3='F7.2,
$ /' Y_4='F7.2,
$ /' YP_1='F7.2,
$ /' YP_3='F7.2,

```

```

$ /'   YP 4=',F7.2,
$ /' SCALE3=',F7.2)
800  FORMAT(11X,'Error Codes'/1X,'-----',/
$ /1X,'DBAR error flag      (DBERR) = ',I1,
$ /1X,'SDAREA error flag   (SDAERR) = ',I1,
$ /1X,'Thrust input error  (T_ERR) = ',I1)
C
C Create useful tables to graph
  IF (PRTFLG) THEN
20    WRITE(6,1100)
    READ *, CHOICE
C
    IF (CHOICE .LT. 0 .OR. CHOICE .GT. 7) THEN
      WRITE(6,*) '*** Invalid Choice ***'
      GO TO 20
    END IF
C
    GO TO (11111, 100, 200, 300, 400, 500, 600, 700) CHOICE+1
C
C Hover ground effect output
100  WRITE(68,*) '1'
    WRITE(68,*) CONFIG,', Hover Ground Effect'
C
    IF (HDEOUT) THEN
      WRITE(68,*) 'H/De'
      DIV = DE_FR
    ELSE
      WRITE(68,*) 'Height'
      DIV = 1.0
    END IF
C
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '6'
    WRITE(68,*) 'OG'
    WRITE(68,*) 'SUCKDOWN'
    WRITE(68,*) 'FOUNTAIN'
    WRITE(68,*) 'LIDS'
    WRITE(68,*) 'TOTAL'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) H
    DO I = 1, H
      WRITE(68,1110) HT(I)/DIV,H_LTOG,H_LTS(I),H_LTF(I),H_LTL(I),
$      H_LT(I),0.0
    END DO
    GO TO 9999
    *****
C
C STOLSF table output
200  WRITE(6,1200)
    READ *, CHOICE
C
    IF (CHOICE .LT. 1 .OR. CHOICE .GT. 3) THEN
      WRITE(6,*) '*** Invalid Choice ***'
      GO TO 200
    END IF
C
    GO TO (210, 230, 220) CHOICE
C
210  WRITE(6,1230)

```

```

DO ID = 1, D
  WRITE(6,1205) ID, DFL(ID)
END DO
WRITE(6,1330)
READ *, ID

C
WRITE(6,1220)
DO IV = 1, V
  WRITE(6,1205) IV, VO(IV)
END DO
WRITE(6,1320)
READ *, IV

C
WRITE(68,*) '1'
WRITE(68,1250) CONFIG,', STOLSF','DFL=',DFL(ID),' VO=', VO(IV)

C
IF (HDEOUT) THEN
  WRITE(68,*) 'H/De'
  DIV = DE_FR
ELSE
  WRITE(68,*) 'Height'
  DIV = 1.0
END IF

C
WRITE(68,*) 'dL/dT'
WRITE(68,*) '4'
WRITE(68,*) 'SUCKDOWN'
WRITE(68,*) 'FOUNTAIN'
WRITE(68,*) 'TOTAL'
WRITE(68,*) 'Zero'
WRITE(68,1105) H
DO I = 1, H
  REC3 = LH*LV*(ID-1) + LH*(IV-1) + I
  READ(70,1109,REC=REC3) SF_LTS
  READ(71,1109,REC=REC3) SF_LTF
  SF_LT = SF_LTS + SF_LTF
  WRITE(68,1110) HT(I)/DIV, SF_LTS, SF_LTF,
    SF_LT, 0.0
$
END DO
GO TO 9999

C
220 WRITE(6,1210)
DO IH = 1, H
  WRITE(6,1205) IH, HT(IH)
END DO
WRITE(6,1310)
READ *, IH

C
WRITE(6,1220)
DO IV = 1, V
  WRITE(6,1205) IV, VO(IV)
END DO
WRITE(6,1320)
READ *, IV

C
WRITE(68,*) '1'
WRITE(68,1250) CONFIG,', STOLSF',' HT=', HT(IH), ' VO=', VO(IV)
WRITE(68,*) 'Jet Deflection Angle'

```

```

WRITE(68,*) 'dL/dT'
WRITE(68,*) '4'
WRITE(68,*) 'SUCKDOWN'
WRITE(68,*) 'FOUNTAIN'
WRITE(68,*) 'TOTAL'
WRITE(68,*) 'Zero'
WRITE(68,1105) D
DO I = 1, D
    REC3 = LH*LV*(I-1) + LH*(IV-1) + IH
    READ(70,1109,REC=REC3) SF_LTS
    READ(71,1109,REC=REC3) SF_LTF
    SF_LT = SF_LTS + SF_LTF
    WRITE(68,1110) DFL(I), SF_LTS, SF_LTF,
$           SF_LT, 0.0
END DO
GO TO 9999

C
230 WRITE(6,1210)
    DO IH = 1, H
        WRITE(6,1205) IH, HT(IH)
    END DO
    WRITE(6,1310)
    READ *, IH

C
    WRITE(6,1230)
    DO ID = 1, D
        WRITE(6,1205) ID, DFL(ID)
    END DO
    WRITE(6,1330)
    READ *, ID

C
    WRITE(68,*) '1'
    WRITE(68,1250) CONFIG,', STOLSF',' HT=', HT(IH), 'DFL=', DFL(ID)

C
    IF (VEOUT) THEN
        WRITE(68,*) 'Ve'
        DIV = SQRT(Q_FR/.00119/1.69**2)
    ELSE
        WRITE(68,*) 'Aircraft Velocity'
        DIV = 1.0
    END IF

C
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '4'
    WRITE(68,*) 'SUCKDOWN'
    WRITE(68,*) 'FOUNTAIN'
    WRITE(68,*) 'TOTAL'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) V
    DO I = 1, V
        REC3 = LH*LV*(ID-1) + LH*(I-1) + IH
        READ(70,1109,REC=REC3) SF_LTS
        READ(71,1109,REC=REC3) SF_LTF
        SF_LT = SF_LTS + SF_LTF
        WRITE(68,1110) VO(I)/DIV, SF_LTS, SF_LTF,
$           SF_LT, 0.0
    END DO
    GO TO 9999

```

```

C *****
C Print-out for STOLGV (Ground vortex term)
300 WRITE(6,1300)
    READ *, CHOICE
C
    IF (CHOICE .LT. 1 .OR. CHOICE .GT. 4) THEN
        WRITE(6,*) '*** Invalid Choice ***'
        GO TO 300
    END IF
C
    GO TO (310, 320, 330, 340) CHOICE
C
C Ground Vortex printout using Altitude as the X-Axis.
310 WRITE(6,1220)
    DO IV = 1, V
        WRITE(6,1205) IV, VO(IV)
    END DO
    WRITE(6,1320)
    READ *, IV
C
    WRITE(6,1230)
    DO ID = 1, D
        WRITE(6,1205) ID, DFL(ID)
    END DO
    WRITE(6,1330)
    READ *, ID
C
    WRITE(6,1240)
    DO IA = 1, A
        WRITE(6,1205) IA, AOA(IA)
    END DO
    READ *, IA
C
    WRITE(68,*) '1'
    WRITE(68,1350) CONFIG,', STOLGV',' VO=', VO(IV),'DFL=',
$      DFL(ID),'AOA=',AOA(IA)
C
    IF (HDEOUT) THEN
        WRITE(68,*) 'H/De'
        DIV = DE_FR
    ELSE
        WRITE(68,*) 'Height'
        DIV = 1.0
    END IF
C
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '4'
    WRITE(68,*) 'Body'
    WRITE(68,*) 'Wing'
    WRITE(68,*) 'Both'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) H
    DO I = 1, H
        REC4 = LH*LV*LD*(IA-1)+LH*LV*(ID-1)+LH*(IV-1)+I
        READ(72,1109,REC=REC4) GV_LTB
        READ(73,1109,REC=REC4) GV_LTW
        GV_LT = GV_LTB + GV_LTW
        WRITE(68,1110) HT(I)/DIV,GV_LTB,

```

```

$                               GV_LTW, GV_LT, 0.0
END DO
GO TO 9999

C
C      Ground Vortex printout using Velocity as the X-Axis.
320  WRITE(6,1210)
      DO IH = 1, H
        WRITE(6,1205) IH, HT(IH)
      END DO
      WRITE(6,1310)
      READ *, IH

C
      WRITE(6,1230)
      DO ID = 1, D
        WRITE(6,1205) ID, DFL(ID)
      END DO
      WRITE(6,1330)
      READ *, ID

C
      WRITE(6,1240)
      DO IA = 1, A
        WRITE(6,1205) IA, AOA(IA)
      END DO
      READ *, IA
      WRITE(68,*) '1'
      WRITE(68,1350) CONFIG,', STOLGV', ' HT=', HT(IH), 'DFL=',
$      DFL(ID), 'AOA=',AOA(IA)

C
      IF (VEOUT) THEN
        WRITE(68,*) 'Ve'
        DIV = SQRT(Q_FR/.00119/1.69**2)
      ELSE
        WRITE(68,*) 'Aircraft Velocity'
        DIV = 1.0
      END IF

C
      WRITE(68,*) 'dL/dT'
      WRITE(68,*) '4'
      WRITE(68,*) 'Body'
      WRITE(68,*) 'Wing'
      WRITE(68,*) 'Both'
      WRITE(68,*) 'Zero'
      WRITE(68,1105) V
      DO I = 1, V
        REC4 = LH*LV*LD*(IA-1)+LH*LV*(ID-1)+LH*(I-1)+IH
        READ(72,1109,REC=REC4) GV_LTB
        READ(73,1109,REC=REC4) GV_LTW
        GV_LT = GV_LTB + GV_LTW
        WRITE(68,1110) VO(I)/DIV, GV_LTB,
$      GV_LTW, GV_LT, 0.0
      END DO
      GO TO 9999

C
C      Ground Vortex printout using jet deflection angle as X-Axis.
330  WRITE(6,1210)
      DO IH = 1, H
        WRITE(6,1205) IH, HT(IH)
      END DO

```

```

WRITE(6,1310)
READ *, IH
C
WRITE(6,1220)
DO IV = 1, V
    WRITE(6,1205) IV, VO(IV)
END DO
WRITE(6,1320)
READ *, IV
C
WRITE(6,1240)
DO IA = 1, A
    WRITE(6,1205) IA, AOA(IA)
END DO
READ *, IA
C
WRITE(68,*) '1'
WRITE(68,1350) CONFIG,', STOLGV',' HT=', HT(IH),' VO=',
$      VO(IV),'AOA=',AOA(IA)
WRITE(68,*) 'Jet deflection angle (Deg)'
WRITE(68,*) 'dL/dT'
WRITE(68,*) '4'
WRITE(68,*) 'Body'
WRITE(68,*) 'Wing'
WRITE(68,*) 'Both'
WRITE(68,*) 'Zero'
WRITE(68,1105) D
DO I = 1, D
    REC4 = LH*LV*LD*(IA-1)+LH*LV*(I-1)+LH*(IV-1)+IH
    READ(72,1109,REC=REC4) GV_LTB
    READ(73,1109,REC=REC4) GV_LTW
    GV_LT = GV_LTB + GV_LTW
    WRITE(68,1110) DFL(I),GV_LTB,
$      GV_LTW, GV_LT, 0.0
END DO
GO TO 9999
C
C
340: Ground Vortex printout using angle of attack as the X-Axis.
WRITE(6,1210)
DO IH = 1, H
    WRITE(6,1205) IH, HT(IH)
END DO
WRITE(6,1310)
READ *, IH
C
WRITE(6,1220)
DO IV = 1, V
    WRITE(6,1205) IV, VO(IV)
END DO
WRITE(6,1320)
READ *, IV
C
WRITE(6,1230)
DO ID = 1, D
    WRITE(6,1205) ID, DFL(ID)
END DO
WRITE(6,1330)
READ *, ID

```

```

C      WRITE(68,*) '1'
      WRITE(68,1350) CONFIG,', STOLGV', ' HT=', HT(IH),
$      ' VO=', VO(IV), 'DFL=', DFL(ID)
      WRITE(68,*) 'AOA (DEG)'
      WRITE(68,*) 'dL/dT'
      WRITE(68,*) '4'
      WRITE(68,*) 'Body'
      WRITE(68,*) 'Wing'
      WRITE(68,*) 'Both'
      WRITE(68,*) 'Zero'
      WRITE(68,1105) A
      DO I = 1, A
        REC4 = LH*LV*LD*(I-1)+LH*LV*(ID-1)+LH*(IV-1)+IH
        READ(72,1109,REC=REC4) GV_LTB
        READ(73,1109,REC=REC4) GV_LTW
        GV_LT = GV_LTB + GV_LTW
        WRITE(68,1110) AOA(I),GV_LTB,
$        GV_LTW, GV_LT, 0.0
      END DO
      GO TO 9999
      *****
C      RCS induced lift loss output
C      400 WRITE(68,*) '1'
      WRITE(68,1360) CONFIG,', RCSIND'
C
      IF (VEOUT) THEN
        WRITE(68,*) 'Ve'
        DIV = SQRT(Q_FR/.00119/1.69**2)
      ELSE
        WRITE(68,*) 'Aircraft Velocity'
        DIV = 1.0
      END IF
C
      WRITE(68,*) 'dL/dT'
      WRITE(68,*) '2'
      WRITE(68,*) 'RCS'
      WRITE(68,*) 'Zero'
      WRITE(68,1105) V
      DO I = 1, V
        WRITE(68,1110) VO(I)/DIV, RCS_LT(1,I), 0.0
      END DO
C
      IF (RCSFLG) THEN
        WRITE(6,1400)
      END IF
C
      GO TO 9999
      *****
C      STOLWK table output
C      500 WRITE(6,1500)
      READ *, CHOICE
C
      IF (CHOICE .LT. 1 .OR. CHOICE .GT. 3) THEN
        WRITE(6,*) '*** Invalid Choice ***'C
        GO TO 500
      END IF
C

```



```

510      GO TO (510, 520, 530) CHOICE
        WRITE(6,1230)
        DO ID = 1, D
            WRITE(6,1205) ID, DFL(ID)
        END DO
        WRITE(6,1330)
        READ *, ID

C
        WRITE(6,1220)
        DO IV = 1, V
            WRITE(6,1205) IV, VO(IV)
        END DO
        WRITE(6,1320)
        READ *, IV

C
        WRITE(68,*) '1'
        WRITE(68,1550) CONFIG,', STOLWK','DFL=',DFL(ID),' VO=', VO(IV)

C
        IF (HDEOUT) THEN
            WRITE(68,*) 'H/De'
            DIV = DE_FR
        ELSE
            WRITE(68,*) 'Height'
            DIV = 1.0
        END IF

C
        WRITE(68,*) 'dL/dT'
        WRITE(68,*) '4'
        WRITE(68,*) 'BODY'
        WRITE(68,*) 'WING'
        WRITE(68,*) 'TOTAL'
        WRITE(68,*) 'Zero'
        WRITE(68,1105) H
        DO I = 1, H
            REC3 = LH*LV*(ID-1) + LH*(IV-1) + I
            READ(74,1109,REC=REC3) WK_LTB
            READ(75,1109,REC=REC3) WK_LTW
            WK_LT = WK_LTB + WK_LTW
            WRITE(68,1110) HT(I)/DIV,WK_LTB, WK_LTW,
                WK_LT, 0.0
        END DO
        GO TO 9999

C
520      WRITE(6,1210)
        DO IH = 1, H
            WRITE(6,1205) IH, HT(IH)
        END DO
        WRITE(6,1310)
        READ *, IH

C
        WRITE(6,1230)
        DO ID = 1, D
            WRITE(6,1205) ID, DFL(ID)
        END DO
        WRITE(6,1330)
        READ *, ID

C
        WRITE(68,*) '1'

```

```

WRITE(68,1550) CONFIG,', STOLWK',' HT=', HT(IH),'DFL=', DFL(ID)
C
IF (VEOUT) THEN
  WRITE(68,*) 'Ve'
  DIV = SQRT(Q_FR/.00119/1.69**2)
ELSE
  WRITE(68,*) 'Aircraft Velocity'
  DIV = 1.0
END IF
C
WRITE(68,*) 'dL/dT'
WRITE(68,*) '4'
WRITE(68,*) 'BODY'
WRITE(68,*) 'WING'
WRITE(68,*) 'TOTAL'
WRITE(68,*) 'Zero'
WRITE(68,1105) V
DO I = 1, V
  REC3 = LH*LV*(ID-1) + LH*(I-1) + IH
  READ(74,1109,REC=REC3) WK_LTB
  READ(75,1109,REC=REC3) WK_LTW
  WK_LT = WK_LTB + WK_LTW
  WRITE(68,1110) VO(I)/DIV, WK_LTB, WK_LTW,
    WK_LT, 0.0
$
END DO
GO TO 9999
C
530 WRITE(6,1210)
DO IH = 1, H
  WRITE(6,1205) IH, HT(IH)
END DO
WRITE(6,1310)
READ *, IH
C
WRITE(6,1220)
DO IV = 1, V
  WRITE(6,1205) IV, VO(IV)
END DO
WRITE(6,1320)
READ *, IV
C
WRITE(68,*) '1'
WRITE(68,1550) CONFIG,', STOLWK',' HT=', HT(IH), ' VO=', VO(IV)
WRITE(68,*) 'Jet Deflection Angle'
WRITE(68,*) 'dL/dT'
WRITE(68,*) '4'
WRITE(68,*) 'BODY'
WRITE(68,*) 'WING'
WRITE(68,*) 'TOTAL'
WRITE(68,*) 'Zero'
WRITE(68,1105) D
DO I = 1, D
  REC3 = LH*LV*(I-1) + LH*(IV-1) + IH
  READ(74,1109,REC=REC3) WK_LTB
  READ(75,1109,REC=REC3) WK_LTW
  WK_LT = WK_LTB + WK_LTW
  WRITE(68,1110) DFL(I),WK_LTB, WK_LTW,
    WK_LT, 0.0
$

```

```

END DO
GO TO 9999
*****
C
C Totals table output
600 WRITE(6,1600)
    READ *, CHOICE
C
    IF (CHOICE .LT. 1 .OR. CHOICE .GT. 4) THEN
        WRITE(6,*) '*** Invalid Choice ***'
        GO TO 600
    END IF
C
    GO TO (610, 620, 630, 640) CHOICE
610 WRITE(6,1220)
    DO IV = 1, V
        WRITE(6,1205) IV, VO(IV)
    END DO
    WRITE(6,1320)
    READ *, IV
C
    WRITE(6,1230)
    DO ID = 1, D
        WRITE(6,1205) ID, DFL(ID)
    END DO
    WRITE(6,1330)
    READ *, ID
C
    WRITE(6,1240)
    DO IA = 1, A
        WRITE(6,1205) IA, AOA(IA)
    END DO
    WRITE(6,1340)
    READ *, IA
C
    WRITE(68,*) '1'
    WRITE(68,1650) CONFIG,', TOTALS', ' VO=', VO(IV), 'DFL=', DFL(ID),
    'AOA=', AOA(IA)
$
C
    IF (HDEOUT) THEN
        WRITE(68,*) 'H/De'
        DIV = DE_FR
    ELSE
        WRITE(68,*) 'Height'
        DIV = 1.0
    END IF
C
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '7'
    WRITE(68,*) 'OGE'
    WRITE(68,*) 'SF'
    WRITE(68,*) 'G-VORTEX'
    WRITE(68,*) 'JET WAKE'
    WRITE(68,*) 'RCS'
    WRITE(68,*) 'TOTAL'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) H
    DO I = 1, H
        REC3 = LH*LV*(ID-1) + LH*(IV-1) + I

```

```

REC4 = LH*LV*LD*(IA-1)+LH*LV*(ID-1)+LH*(IV-1)+I
READ(70,1109,REC=REC3) SF_LTS
READ(71,1109,REC=REC3) SF_LTF
READ(72,1109,REC=REC4) GV_LTB
READ(73,1109,REC=REC4) GV_LTW
READ(74,1109,REC=REC3) WK_LTB
READ(75,1109,REC=REC3) WK_LTW
SF_LT = SF_LTS + SF_LTF
GV_LT = GV_LTB + GV_LTW
WK_LT = WK_LTB + WK_LTW
TOT = HLTOGE(ID) + SF_LT + GV_LT +
      WK_LT + RCS_LT(1,IV)
$   WRITE(68,1110) HT(I)/DIV, HLTOGE(ID), SF_LT,
$   GV_LT, WK_LT, RCS_LT(1,IV),
$   TOT, 0.0
END DO
GO TO 9999
C
620 WRITE(6,1210)
    DO IH = 1, H
        WRITE(6,1205) IH, HT(IH)
    END DO
    WRITE(6,1310)
    READ *, IH
C
    WRITE(6,1230)
    DO ID = 1, D
        WRITE(6,1205) ID, DFL(ID)
    END DO
    WRITE(6,1330)
    READ *, ID
C
    WRITE(6,1240)
    DO IA = 1, A
        WRITE(6,1205) IA, AOA(IA)
    END DO
    WRITE(6,1340)
    READ *, IA
C
    WRITE(68,*) '1'
    WRITE(68,1650) CONFIG,', TOTALS',' HT=', HT(IH), 'DFL=', DFL(ID),
$   'AOA=', AOA(IA)
C
    IF (VEOUT) THEN
        WRITE(68,*) 'Ve'
        DIV = SQRT(Q_FR/.00119/1.69**2)
    ELSE
        WRITE(68,*) 'Aircraft Velocity'
        DIV = 1.0
    END IF
C
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '7'
    WRITE(68,*) 'OGE'
    WRITE(68,*) 'SF'
    WRITE(68,*) 'G-VORTEX'
    WRITE(68,*) 'JET WAKE'
    WRITE(68,*) 'RCS'

```

```

WRITE(68,*) 'TOTAL'
WRITE(68,*) 'Zero'
WRITE(68,1105) V
DO I = 1, V
  REC3 = LH*LV*(ID-1) + LH*(I-1) + IH
  REC4 = LH*LV*LD*(IA-1)+LH*LV*(ID-1)+LH*(I-1)+IH
  READ(70,1109,REC=REC3) SF_LTS
  READ(71,1109,REC=REC3) SF_LTF
  READ(72,1109,REC=REC4) GV_LTB
  READ(73,1109,REC=REC4) GV_LTW
  READ(74,1109,REC=REC3) WK_LTB
  READ(75,1109,REC=REC3) WK_LTW
  SF_LT = SF_LTS + SF_LTF
  GV_LT = GV_LTB + GV_LTW
  WK_LT = WK_LTB + WK_LTW
  TOT = HLTOGE(ID) + SF_LT + GV_LT +
    WK_LT + RCS_LT(1,I)
$   WRITE(68,1110) VO(I)/DIV, HLTOGE(ID), SF_LT,
$   GV_LT, WK_LT, RCS_LT(1,I),
$   TOT, 0.0
END DO
GO TO 9999

C
630 WRITE(6,1210)
    DO IH = 1, H
      WRITE(6,1205) IH, HT(IH)
    END DO
    WRITE(6,1310)
    READ *, IH

C
    WRITE(6,1220)
    DO IV = 1, V
      WRITE(6,1205) IV, VO(IV)
    END DO
    WRITE(6,1320)
    READ *, IV

C
    WRITE(6,1240)
    DO IA = 1, A
      WRITE(6,1205) IA, AOA(IA)
    END DO
    WRITE(6,1340)
    READ *, IA

C
    WRITE(68,*) '1'
    WRITE(68,1650) CONFIG,', TOTALS', ' HT=', HT(IH), ' VO=', VO(IV),
$      'AOA=', AOA(IA)
    WRITE(68,*) 'Jet Deflection Angle'
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '7'
    WRITE(68,*) 'OGE'
    WRITE(68,*) 'SF'
    WRITE(68,*) 'G-VORTEX'
    WRITE(68,*) 'JET WAKE'
    WRITE(68,*) 'RCS'
    WRITE(68,*) 'TOTAL'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) D

```

```

DO I = 1, D
  REC3 = LH*LV*(I-1) + LH*(IV-1) + IH
  REC4 = LH*LV*LD*(IA-1)+LH*LV*(I-1)+LH*(IV-1)+IH
  READ(70,1109,REC=REC3) SF_LTS
  READ(71,1109,REC=REC3) SF_LTF
  READ(72,1109,REC=REC4) GV_LTB
  READ(73,1109,REC=REC4) GV_LTW
  READ(74,1109,REC=REC3) WK_LTB
  READ(75,1109,REC=REC3) WK_LTW
  SF_LT = SF_LTS + SF_LTF
  GV_LT = GV_LTB + GV_LTW
  WK_LT = WK_LTB + WK_LTW
  TOT = HLTOGE(I) + SF_LT + GV_LT +
$      WK_LT + RCS_LT(1,IV)
  WRITE(68,1110) DFL(I), HLTOGE(I), SF_LT,
$      GV_LT, WK_LT, RCS_LT(1,IV),
$      TOT, 0.0
END DO
GO TO 9999

C
640 WRITE(6,1210)
    DO IH = 1, H
      WRITE(6,1205) IH, HT(IH)
    END DO
    WRITE(6,1310)
    READ *, IH

C
    WRITE(6,1220)
    DO IV = 1, V
      WRITE(6,1205) IV, VO(IV)
    END DO
    WRITE(6,1320)
    READ *, IV

C
    WRITE(6,1230)
    DO ID = 1, D
      WRITE(6,1205) ID, DFL(ID)
    END DO
    WRITE(6,1320)
    READ *, ID

C
    WRITE(68,*) '1'
    WRITE(68,1650) CONFIG,', TOTALS', ' HT=', HT(IH), ' VO=', VO(IV),
$      'DFL=', DFL(ID)
    WRITE(68,*) 'Aircraft Angle of Attack'
    WRITE(68,*) 'dL/dT'
    WRITE(68,*) '7'
    WRITE(68,*) 'OGE'
    WRITE(68,*) 'SF'
    WRITE(68,*) 'G-VORTEX'
    WRITE(68,*) 'JET WAKE'
    WRITE(68,*) 'RCS'
    WRITE(68,*) 'TOTAL'
    WRITE(68,*) 'Zero'
    WRITE(68,1105) A
    DO I = 1, A
      REC3 = LH*LV*(ID-1) + LH*(IV-1) + IH
      REC4 = LH*LV*LD*(I-1)+LH*LV*(ID-1)+LH*(IV-1)+IH

```

```

      READ(70,1109,REC=REC3) SF_LTS
      READ(71,1109,REC=REC3) SF_LTF
      READ(72,1109,REC=REC4) GV_LTB
      READ(73,1109,REC=REC4) GV_LTW
      READ(74,1109,REC=REC3) WK_LTB
      READ(75,1109,REC=REC3) WK_LTW
      SF_LT = SF_LTS + SF_LTF
      GV_LT = GV_LTB + GV_LTW
      WK_LT = WK_LTB + WK_LTW
      TOT = HLTOGE(ID) + SF_LT + GV_LT +
$      WK_LT + RCS_LT(1,IV)
      WRITE(68,1110) HT(I)/DIV, HLTOGE(ID), SF_LT,
$      GV_LT, WK_LT, RCS_LT(1,IV),
$      TOT, 0.0
      END DO
      GO TO 9999
*****
C      JETIND table output
C      WRITE(6,1240)
700      DO IA = 1, A
          WRITE(6,1205) IA, AOA(IA)
      END DO
      WRITE(6,1320)
      READ *, IA
C
      CALL JIETAB(IA)
C
9999      CLOSE (68)
          GO TO 20
      END IF
C
1100      FORMAT(15X,'OUTPUT DATA TO TABLE FORMAT'
$      ///5X,'0 - Quit table generation section'
$      //5X,'1 - Output table with all hover results (dL/dT vs Height)'
$      //5X,'2 - Tables with results from STOLSF (any)'
$      //5X,'3 - Tables with results from STOLGV (any)'
$      //5X,'4 - Table with results from RCSIND'
$      //5X,'5 - Table with results from STOLWK (any)'
$      //5X,'6 - Table with totals from all routines'
$      //5X,'7 - Table for JETIND'
$      //5X,'Choose an option')
1105      FORMAT (1X,I4)
1109      FORMAT (F11.5)
1110      FORMAT(1X,F8.4,3X,14(F11.5,2X))
C
1200      FORMAT(1X,'* * * * *')
$      /15X,'STOLSF Table generation section (Suckdown/Fountain)',
$      ///5X,'1 - Output table based on height',
$      //5X,'2 - Output table based on aircraft velocity'
$      //5X,'3 - Output table based on jet deflection angle',
$      ///5X,'Choose an option')
1205      FORMAT (5X,I3,' --> ',F6.1)
1210      FORMAT(//3X,'H Number',4X,'Height')
1220      FORMAT(//3X,'V Number',4X,'Velocity')
1230      FORMAT(//3X,'D Number',4X,'Def Angle')
1240      FORMAT(//3X,'A Number',4X,'Angle of Attack')
1250      FORMAT(1X,A18,A8,2(2X,A4,F3.0))
C

```

```

1300  FORMAT(1X,'* * * * *')
      $ /15X,'STOLGV Table generation section (Ground Vortex)'/
      $ //5X,'1 - Output table based on height',
      $ //5X,'2 - Output table based on aircraft velocity',
      $ //5X,'3 - Output table based on jet deflection angle',
      $ //5X,'4 - Output table based on aircraft angle of attack',
      $ ///5X,'Choose an option')
1310  FORMAT(1X,'Enter your choice for Height (H Number)')
1320  FORMAT(1X,'Enter your choice for Aircraft Velocity (V Number)')
1330  FORMAT(1X,'Enter your choice for Deflection Angle (D Number)')
1340  FORMAT(1X,'Enter your choice for Aircraft Angle of Attack
      $ (A Number)')
1350  FORMAT(1X,A18,A8,3(2X,A4,F3.0))
1360  FORMAT(1X,A18,A8)
C
1400  FORMAT(5X,'*****')
      $ 5X,'* RCS table output could be incorrect *'/
      $ 5X,'* because the ratio of wing area over *'/
      $ 5X,'* jet area has exceeded 7000. *'/
      $ 5X,'*****')
C
1500  FORMAT(1X,'* * * * *')
      $ /15X,'STOLWK Table generation section (jet WaKe)',
      $ ///5X,'1 - Output table based on height',
      $ //5X,'2 - Output table based on aircraft velocity',
      $ //5X,'3 - Output table based on jet deflection angle',
      $ ///5X,'Choose an option')
1550  FORMAT(1X,A18,A8,2(2X,A4,F3.0))
C
1600  FORMAT(1X,'* * * * *')
      $ /15X,'Table generation for Totals',
      $ ///5X,'1 - Output table based on height',
      $ //5X,'2 - Output table based on aircraft velocity',
      $ //5X,'3 - Output table based on jet deflection angle',
      $ //5X,'4 - Output table based on Aircraft Angle of Attack',
      $ ///5X,'Choose an option')
1650  FORMAT(1X,A18,A8,3(2X,A3,F3.0))
11111 RETURN
      END

```

Dist

```

      FUNCTION DIST(X1,Y1,X2,Y2)
C * * * * *
C THIS FUNCTION CALCULATES THE DISTANCE BETWEEN THE TWO GIVEN
C COORDINATES IN THE (X,Y) PLANE.
C
C   GIVEN:  TWO POINTS (X1, Y1), (X2, Y2)
C   FIND:   THE DISTANCE BETWEEN THE TWO POINTS
C
C   REAL BLEEP, DIST, X1, Y1, X2, Y2
C
C   BLEEP = (X2-X1)**2 + (Y2-Y1)**2
C   DIST = SQRT(BLEEP)
C
C   RETURN
      END

```


Perpdis

```

SUBROUTINE PERPDIS(X,Y,MLINE,BLINE,dist)
C * * * * *
C THIS SUBROUTINE CALCULATES THE DISTANCE FROM A POINT TO A LINE IN A
C PARTICULAR DIRECTION DEFINED BY A SLOPE.
C
C   REAL MLINE, a,b,c,x,y, dist
C
C   A = -MLINE
C   B = 1.0
C   C = -BLINE
C
C   CALCULATE PERPENDICULAR DISTANCE BETWEEN POINT AND LINE
C   DIST = (A * X + B * Y + C)/SQRT(A**2 + B)
C
C   RETURN
C   END

```

Linterp

```

SUBROUTINE LINTERP (X,X1,X2,Y1,Y2,Y)
C * * * * *
C PURPOSE: THIS SUBROUTINE PREFORMS LINEAR INTERPOLATION.
C GIVEN: TWO POINTS (X1,Y1), (X2,Y2)
C        A VALUE (X) ON THE X-AXIS BETWEEN THE TWO POINTS
C FIND: A VALUE (Y) THAT CORRESPONDS TO THE VALUE (X) THAT LIES
C       ALONG THE LINE DEFINED BY THE TWO POINTS.
C
C   REAL X,X1,X2,Y,Y1,Y2
C
C   Y = Y1 + (Y2-Y1)/(X2-X1)*(X-X1)
C
C   RETURN
C   END

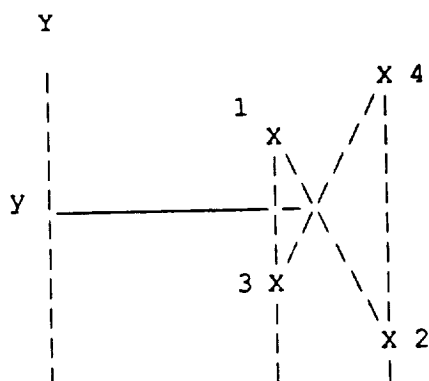
```

Crossin

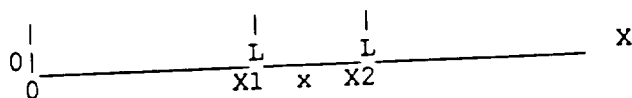
```

SUBROUTINE CROSSIN(X1,Y1,X2,Y2,X3,Y3,X4,Y4,x,y)
C * * * * *
C THIS SUBROUTINE CALCULATES THE INTERSECTION OF TWO LINES, EACH LINE
C DEFINED BY TWO POINTS. EACH OF THE LINES END-POINTS LIE ON THE SAME
C X-COORDINATE
C
C   GIVEN: FOUR POINTS THAT DEFINE TWO CROSSING LINES (AS SHOWN)

```



C
C
C
C
C
C
C



FIND: THE INTERSECTION OF THE TWO LINES (x,y)

REAL X1,Y1,X2,Y2,X3,Y3,X4,Y4,x,y, SLOPE1, SLOPE2, B1, B2

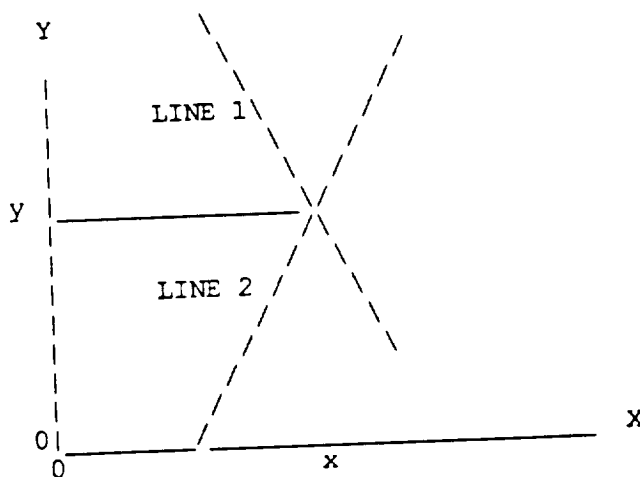
SLOPE1 = (Y2 - Y1)/(X2 - X1)
SLOPE2 = (Y4 - Y3)/(X4 - X3)
B1 = -X1*SLOPE1 + Y1
B2 = -X3*SLOPE2 + Y3
CALL LINECRO(SLOPE1,B1,SLOPE2,B2,x,y)
RETURN
END

Linecro

SUBROUTINE LINECRO(SLOPE1,B1,SLOPE2,B2,x,y)

C * * * * *
C THIS SUBROUTINE CALCULATES THE INTERSECTION OF TWO LINES

GIVEN: SLOPE1 = SLOPE OF THE FIRST LINE
B1 = THE Y-INTERCEPT OF THE FIRST LINE
SLOPE2 = SLOPE OF THE SECOND LINE
B2 = THE Y-INTERCEPT OF THE SECOND LINE



FIND: THE INTERSECTION OF THE TWO LINES (x,y)

REAL SLOPE1,B1,SLOPE2,B2,x,y

Y = SLOPE2*(B2-B1)/(SLOPE1 - SLOPE2) + B2

IF (SLOPE1 .EQ. 0.0) THEN
X = (Y - B2)/SLOPE2
ELSE
X = (Y - B1)/SLOPE1
END IF

RETURN
END

C

Polyar

SUBROUTINE POLYAR

```

C-----
C ACRONYM:  POLYgon Area
C -----
C PURPOSE:  The purpose of POLYAR is to calculate the area of a ploygon
C           by defined a number of points.
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   I          I      ---      ---      General purpose counter
C   SUM        R      I      ---      Portion added to total
C   SUB        R      I      ---      Portion subtracted from total
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C   NAME      TYPE  I/O  UNITS      DESCRIPTION
C   -----
C   POINTS     I      I      ---      Number of points(not to exceed 500)
C   XPTS(500)  R      I      ---      X-coordinates of polygon
C   YPTS(500)  R      I      ---      Y-coordinates of polygon
C   TOTAL      R      I      ---      Total area enclosed by the polygon.
C FILES USED:
C   LOGICAL UNIT  I/O  DESCRIPTION
C   -----
C   (none)
C COMMONS USED:
C   NAME          DESCRIPTION
C   -----
C   POLYGON       Contains the points used in POLYAR when calculating the
C                 area of a polygon and CENTAR when calculating the center
C                 of area.
C CALLED BY:
C   NAME          DESCRIPTION
C   -----
C   FINVAR        Calculates all variables which have not been defined by
C                 the user or set to a default value
C   CENTAR        Calculates the center of area, area and area in front of
C                 a point for a given planform
C ROUTINES CALLED:
C   NAME          DESCRIPTION
C   -----
C   (none)
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   None.
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Res Cntr
C
C REVISION HISTORY:
C   DATE          INITIALS & DESCRIPTION
C   05/15/90     KEH -- Initial completion of code.
C   07/14/91     KEH -- Eliminated extraneous spacing and fixed comments
C-----
#include "misc.inc"
C

```

```

C Variable declarations
  REAL SUB, SUM
  INTEGER I

C
  SUM = 0.0
  SUB = 0.0
  DO 10 I = 1, POINTS

C
    IF (I .EQ. POINTS) THEN
      SUM = SUM + XPTS(I) * YPTS(1)
      SUB = SUB + YPTS(I) * XPTS(1)
    ELSE
      SUM = SUM + XPTS(I)*YPTS(I+1)
      SUB = SUB + YPTS(I)*XPTS(I+1)
    END IF

C
10  CONTINUE
    TOTAL = ABS(.5 * (SUM - SUB))

C
  RETURN
  END

```

Centar

SUBROUTINE CENTAR(MODE,XSTART,X,Y,N,OUT)

```

C-----
C ACRONYM:  CENTER of Area
C -----
C PURPOSE:  The purpose of CENTAR is to calculate a planform center of
C           area based on the planform defined by X and Y.
C
C PARAMETERS:
C   NAME      TYPE  I/O  UNITS  DESCRIPTION
C   -----
C   MODE      I     I    ----   Flag which tells this routine which
C                                   values to calculate
C                                   1 - Calculate Center of Area
C                                   2 - Calculate Area of planform
C                                   3 - Calculate area in front of
C                                   XSTART
C   N          I     I    ----   Number of points in planform
C   OUT        R     O    ----   Value to be returned to calling
C                                   routine. Based on MODE.
C   X(500)     R     I    ----   X-coordinate of planform
C   XSTART     R     I    ----   Point from which to calculate the
C                                   area forward of this point
C   Y(500)     R     I    ----   Y-coordinate of planform
C LOCAL VARIABLES (in addition to the above parameters):
C   NAME      TYPE  I/O  UNITS  DESCRIPTION
C   -----
C   ALEFT     R          ----   Left side Area difference for
C                                   bisection method
C   ARIGHT    R          ----   Right side Area difference for
C                                   bisection method
C   I          I          ----   General purpose counter
C   TEST      R          ----   Test Area difference for bisection
C                                   method
C   X_CA      R     I    ----   X-coordinate of Centr of Area
C   XLEFT     R          ----   Left side X-coordinate for

```

```

C                                     Bisection method
C XRIGHT      R      ----      Right side X-coordinate for
C                                     bisection method.
C YPT         R      ----      Y-coordinate of on planform that is
C                                     directly above the guess of the
C                                     center of area
C GLOBAL VARIABLES (in addition to the above parameters and local vars):
C   NAME      TYPE  I/O  UNITS  DESCRIPTION
C   -----
C   POINTS     I     I   ----   Number of points(not to exceed 500)
C   XPTS(500)  R     I   ----   X-coordinates of polygon
C   YPTS(500)  R     I   ----   Y-coordinates of polygon
C FILES USED:
C   LOGICAL UNIT  I/O  DESCRIPTION
C   -----
C   (none)
C COMMONS USED:
C   NAME          DESCRIPTION
C   -----
C   POLYGON       Contains the points used in POLYAR when calculating the
C                   area of a polygon and CENTAR when calculating the center
C                   of area.
C CALLED BY:
C   NAME          DESCRIPTION
C   -----
C   FINVAR        Calculates all variables which have not been defined by
C                   the user or set to a default value
C ROUTINES CALLED:
C   NAME          DESCRIPTION
C   -----
C   LINTERP       Linear interpolates between two X and Y values give an
C                   intermediate X value
C   PERPDIS       Calculates the perpendicular distance between a point
C                   and a line
C   POLYAR        Calculates the area of any polygon
C NOTES: None.
C REFERENCES:
C   1) None.
C ENVIRONMENT:
C   FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C   None.
C AUTHOR(S):
C   Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Moffett
C REVISION HISTORY:
C   DATE          INITIALS & DESCRIPTION
C   04/01/90      KEH -- Code completed and in working order
C   07/14/91      KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "misc.inc"
C
C Variable declarations
C   REAL X CA, X(500), Y(500)
C   INTEGER I, N, FIRST
C
C   FIRST = 1
C Calculate the geometric center of the planform.
C   GEOCTR = (X(N) + X(1)) / 2.0

```

```

C
C Initialize X_CA based on the value of XSTART
  IF (XSTART .EQ. 9999.) THEN
    X_CA = GEOCTR
  ELSE
    X_CA = XSTART
  END IF

C
C Start calculating areas
1 DO 20 I = 2, N
C
  IF (X_CA .LE. X(I) .AND. X_CA .GT. X(I-1)) THEN
    CALL LINTERP(X_CA, X(I-1), X(I), Y(I-1), Y(I), YPT)
    Set up forward half of planform to calculate its area
    DO 5 J = 1, I-1
      XPTS(J) = X(J)
      YPTS(J) = Y(J)
5    CONTINUE
      XPTS(I) = X_CA
      YPTS(I) = YPT
      XPTS(I+1) = X_CA
      YPTS(I+1) = 0.0
      POINTS = I+1
C      Calculate front area
      CALL POLYAR
      FAREA = TOTAL
C      Set up rear half of planform to calculate its area.
      XPTS(1) = X_CA
      YPTS(1) = 0.0
      XPTS(2) = X_CA
      YPTS(2) = YPT
      DO 10 K = I, N
        J = K - I + 3
        XPTS(J) = X(K)
        YPTS(J) = Y(K)
10     CONTINUE
        POINTS = J
C      Calculate rear area
      CALL POLYAR
      RAREA = TOTAL
    END IF
  END IF

20 CONTINUE

C
C Initialize left and right side for bisection method
C
  IF (FIRST .EQ. 1) THEN
    TAREA = RAREA + FAREA
    ALEFT = 0.0 - TAREA
    XLEFT = X(1)
    ARIGHT = TAREA - 0.0
    XRIGHT = X(N)
    FIRST = 2
  END IF

C
C Set OUT to the correct value according to the value of MODE.
  IF (MODE .EQ. 2) THEN
    OUT = TAREA
    GO TO 60
  
```

```

ELSE IF (MODE .EQ. 3) THEN
  OUT = FAREA
  GO TO 60
END IF

C
TEST = FAREA - RAREA

C
C Determine which side TEST is on (Right or Left).
IF (TEST .LT. 0.0) THEN
  ALEFT = TEST
  XLEFT = X_CA
ELSE
  ARIGHT = TEST
  XRIGHT = X_CA
END IF

C
C Check to see if Areas are within .5% of each other
C (for Center of area calculations)
C IF (ABS(TEST/(TAREA/2.0)) .LT. .005) GO TO 50
X_CA = (XLEFT + XRIGHT)/2.0
C Return to calculate new FAREA and RAREA based on the new X_CA
GO TO 1

C
50 IF (MODE .EQ. 1) OUT = X_CA
C
60 RETURN
END

```

Ssen

SUBROUTINE SSEN(START, STEP, END, NUM, DSTART, DEND, DNUM, FLAG)

C-----
C ACRONYM: Start, Step, End, Number calculator

C
C PURPOSE: The purpose of SSEN is to calculate any of the variables
C that define the number list which are not defined, using
C default values when not enough information is given.

C PARAMETERS:

NAME	TYPE	I/O	UNITS	DESCRIPTION
DEND	R	I	----	Default value for ending value
DNUM	I	I	----	Default value for number of points
DSTART	R	I	----	Default value for starting value
END	R	I	----	Ending value of number list
FLAG	R	I	----	Flag to which values are set when the values are not defined. (When setting NUM as undefined use the integer equivalent of FLAG.)
NUM	I	I	----	Number of points in number list
START	R	I	----	Starting value of number list
STEP	R	I	----	Stepping value of number list

C LOCAL VARIABLES (in addition to the above parameters):

NAME	TYPE	I/O	UNITS	DESCRIPTION
(none)				

C GLOBAL VARIABLES (in addition to the above parameters and local vars):

NAME	TYPE	I/O	UNITS	DESCRIPTION

```

C (none)
C FILES USED:
C LOGICAL UNIT I/O DESCRIPTION
C -----
C (none)
C
C COMMONS USED:
C NAME DESCRIPTION
C -----
C (none)
C CALLED BY:
C NAME DESCRIPTION
C -----
C FINVAR Calculates all variables which have not been defined by
C the user or set to a default value
C
C ROUTINES CALLED:
C NAME DESCRIPTION
C -----
C (none)
C NOTES: None.
C REFERENCES:
C 1) None.
C ENVIRONMENT:
C FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C None.
C AUTHOR(S):
C Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Res Cntr
C REVISION HISTORY:
C DATE INITIALS & DESCRIPTION
C 07/18/90 KEH -- Code completed and in working order
C 07/14/91 KEH -- Eliminated extraneous spacing and fixed comments
C -----
C Variable Declaration
C REAL START, STEP, END, DSTART, DEND
C INTEGER DNUM, NUM
C
C IF (START .NE. FLAG) THEN
C
C IF (END .NE. FLAG) THEN
C
C IF (STEP .NE. FLAG) THEN
C NUM = INT((END - START) / STEP + 1.)
C ELSE IF (NUM .NE. INT(FLAG)) THEN
C
C IF (NUM .EQ. 1) THEN
C STEP = 0.0
C ELSE
C STEP = (END - START) / (NUM - 1)
C END IF
C
C ELSE
C NUM = DNUM
C
C IF (NUM .EQ. 1) THEN
C STEP = 0.0
C ELSE
C STEP = (END - START) / (NUM - 1)

```



```

      END IF
C
      END IF
C
      ELSE IF (STEP .NE. FLAG) THEN
C
        IF (NUM .NE. FLAG) THEN
          END = START + STEP * (NUM - 1)
        ELSE
          END = DEND
          NUM = INT((END - START) / STEP + 1.)
        END IF
C
      ELSE IF (NUM .NE. FLAG) THEN
        END = DEND
C
        IF (NUM .EQ. 1) THEN
          STEP = 0.0
        ELSE
          STEP = (END - START) / (NUM - 1)
        END IF
C
      ELSE
        END = DEND
        NUM = DNUM
C
        IF (NUM .EQ. 1) THEN
          STEP = 0.0
        ELSE
          STEP = (END - START) / (NUM - 1)
        END IF
C
      END IF
C
      ELSE IF (END .NE. FLAG) THEN
C
        IF (STEP .NE. FLAG) THEN
C
          IF (NUM .NE. INT(FLAG)) THEN
            START = END - STEP * (NUM - 1)
          ELSE
            START = DSTART
            NUM = INT((END - START) / STEP + 1.)
          END IF
C
        ELSE IF (NUM .NE. FLAG) THEN
          START = DSTART
          STEP = (END - START) / (NUM - 1)
        ELSE
          START = DSTART
          NUM = DNUM
          STEP = (END - START) / (NUM - 1)
        END IF
C
      ELSE IF (STEP .NE. FLAG) THEN
C
        IF (NUM .NE. FLAG) THEN
          START = DSTART

```

```

      END = START + STEP * (NUM - 1)
    ELSE
      START = DSTART
      END = DEND
      NUM = INT((END - START) / STEP + 1.)
    END IF

C
    ELSE IF (NUM .NE. FLAG) THEN
      START = DSTART
      END = DEND
      STEP = (END - START) / (NUM - 1)
    ELSE
      START = DSTART
      END = DEND
      NUM = DNUM
      STEP = (END - START) / (NUM - 1)
    END IF

C
    RETURN
  END

```

Jietab

SUBROUTINE JIETAB(L)

C-----
 C ACRONYM: Jet Induced Effect TABLE generator - Example table follows:

```

C
C      #_of_Heights #_of_Velocities #_of_Deflection_Angles
C      --> DFL1 Vel1 Vel2 .... VelN
C          Lt1  LT11 LT12 .... Lt1N
C          Lt2  LT21 LT22 .... Lt2N
C          :    :    :
C          :    :    :
C          LtM  LTM1 LTM2 .... LTMN
C      --> DFL2 Vel1 Vel2 .... VelN
C          Lt1  LT11 LT12 .... Lt1N
C          :    :    :
C          etc. etc. etc.    etc.
C      - - - - -
C

```

C PURPOSE: The purpose of JIETAP is to generate the table which JETIND
 C uses to determine the lift loss

C PARAMETERS:

NAME	TYPE	I/O	UNITS	DESCRIPTION
L	I	I	----	Angle of attack which table is to be generated

C LOCAL VARIABLES (in addition to the above parameters):

NAME	TYPE	I/O	UNITS	DESCRIPTION
I	I	I	----	Height counter
J	I	I	----	Velocity counter
K	I	I	----	Deflection angle counter
DUMMY(100)	I	I	----	Dummy array variable to hold total results so routine can output values in an implied do loop.

C GLOBAL VARIABLES (in addition to the above parameters and local vars):

NAME	TYPE	I/O	UNITS	DESCRIPTION
------	------	-----	-------	-------------

```

C -----
C -----
C RESPIE Common Block
C NAME TYPE I/O UNITS DESCRIPTION
C -----
C DFL(500) R I deg Actual nozzle deflection angle
C value
C HT(500) R O Ft Actual height value
C LD R I ---- Total number of nozzle deflection
C angles.
C LH R I ---- Total number of height values.
C LV R I ---- Total number of velocity values.
C VO(500) R O ---- Actual velocity value
C FILES USED:
C LOGICAL UNIT I/O DESCRIPTION
C -----
C 68 O Sequential file for table output
C COMMONS USED:
C NAME DESCRIPTION
C -----
C RESPIE RESULTS from all Power Induced Effects subroutines -
C Contains results from each subroutine along with
C variables for height, velocity, jet deflection angle,
C and angle of attack and their counters.
C CALLED BY:
C NAME DESCRIPTION
C -----
C JIETAB Creates output file which is used by ACSYNT
C ROUTINES CALLED:
C NAME DESCRIPTION
C -----
C RTOTAL Sums the total lift increment given array type indices
C NOTES: None.
C REFERENCES:
C 1) None.
C ENVIRONMENT:
C FORTRAN 77, VAX 6800, VAX 11/785, SGI IRIS 4D series.
C NON-STANDARD CODE:
C None.
C AUTHOR(S):
C Kipp E. Howard (KEH), Cal Poly San Luis Obispo, NASA Ames Res Cntr
C REVISION HISTORY:
C DATE INITIALS & DESCRIPTION
C 08/07/90 KEH -- Code completed and in working order
C 07/14/91 KEH -- Eliminated extraneous spacing and fixed comments
C -----
C #include "respie.inc"
C
C REAL RTOTAL, DUMMY(100)
C INTEGER I,J,K,L
C
C Output first line of table (Number of Altitudes, Number of Velocities
C Number of Nozzle Deflections)
C WRITE(68,*) LH, LV, LD
C
C Start Loops
C
C Deflection LoopC

```

```

DO K = 1, LD
  WRITE(68,100) DFL(K), (VO(J), J = 1, LV)
C
C      Altitude Loop
C      DO I = 1, LH
C          Dummy variable has to be used because, for some reason, a
C          function, with READ statements, within an implied do loop
C          will cause the WRITE statement to output nothing. A
C          function within an implied do loop works fine as long as
C          there are no READ statements.
C
C          Velocity Loop
C          DO J = 1, LV
C              DUMMY(J) = RTOTAL(I,J,K,L)
C          END DO
C          WRITE(68,100) HT(I), (DUMMY(J), J = 1, LV)
C      END DO
C  END DO
100  FORMAT(1X,101(G11.5,2X))
C
C      RETURN
C      END
C      REAL FUNCTION RTOTAL(I,J,K,L)
C
C      I      -      Altitude variable
C      J      -      Velocity Variable
C      K      -      Deflection Variable
C      L      -      Angle of Attack Variables
C
C      #include "respie.inc"
C
C      INTEGER I, J, K, L
C
C      REC3 = LH*LV*(K-1) + LH*(J-1) + I
C      REC4 = LH*LV*LD*(L-1)+LH*LV*(K-1)+LH*(J-1)+I
C      READ(70,100,REC=REC3) SF_LTS
C      READ(71,100,REC=REC3) SF_LTF
C      READ(72,100,REC=REC4) GV_LTB
C      READ(73,100,REC=REC4) GV_LTW
C      READ(74,100,REC=REC3) WK_LTB
C      READ(75,100,REC=REC3) WK_LTW
C      SF_LT = SF_LTS + SF_LTF
C      GV_LT = GV_LTB + GV_LTW
C      WK_LT = WK_LTB + WK_LTW
C      RTOTAL = HLTOGE(J) + SF_LT + GV_LT + WK_LT + RCS_LT(1,J)
C      RTOTAL = RTOTAL
C
C      100  FORMAT (F11.5)
C
C      RETURN
C      END

```

APPENDIX B

Power Induced Effects Module User's/Programmer's Manuals

Power Induced Effects Module User's/Programmer's Manuals

by

Kipp Edward Howard

July 1991

Table of Contents

	<u>Page</u>
User's Guide	1
Description	1
Philosophy	1
Control	3
Input	3
Preliminary Calculations	4
Lift Increments Routines	4
Height	4
Forward Velocity	4
Height, Forward Velocity, and Jet Deflection Angle	4
Height, Forward Velocity, Jet Deflection Angle, and Angle of Attack	5
Output	5
Limitations	6
Variables	6
Definitions	23
Minimum Configuration Variables	24
Necessary Group	24
Thrust Group	25
Nozzle Definition Group	25
Pressure Ratio Group	25
RCS Group	26
LIDs Group	26
Potential Group	26
Output	27
Example	27
Configuration	27
Input	29
Execution	34
Output	40
Programmer's Guide	40
Overall Scheme & Assumptions	40
Equations for Subroutines	40
Hover	40
Suckdown	41
Fountain Lift	41
Basic Method	43
h' Method	44
Lift Improvement Devices	45
Forward Flight / Jet Deflection Angle / Angle of Attack	45
Suckdown/Fountain	46
Ground Vortex	46
Jet Wake	46
Reaction Control System	47

Routine Descriptions.....	48
Control.....	48
Preliminary Routines.....	49
INPIE.....	49
PLANMO.....	49
FINVAR.....	50
SDAREA.....	50
DIABAR.....	50
Hover.....	50
HCALL.....	51
HOV_GE.....	51
Suckdown/Fountain.....	51
SFCALL.....	51
STOLSF.....	51
Ground Vortex.....	51
GVCALL.....	51
STOLGV.....	52
Jet Wake.....	52
WKCALL.....	52
JIEPIE.....	52
STOLWK.....	52
Reaction Control System.....	52
RCSCAL.....	52
RCSIND.....	53
Output.....	53
PIEP.....	53
JIETAB.....	53
Miscellaneous.....	53
CENTAR.....	53
CROSSIN.....	53
DIST.....	53
INTEGRA.....	54
LINECRO.....	54
LINTERP.....	54
PERPDIS.....	54
POLYAR.....	54
RTOTAL.....	54
SSEN.....	54
References.....	55
Appendices.....	56
Routine Summary.....	56
PIE Hierarchy Diagram.....	58
Cross Reference - Common Blocks.....	59
Cross Reference - Calls To.....	60
Cross Reference - Calls By.....	61

User's Guide

This manual will provide the user of the Power Induced Effects Module with enough information to implement PIE and provide the programmer with adequate information to support and/or enhance the PIE code

This work was performed by Kipp Edward Howard for his Masters thesis at Cal Poly San Luis Obispo as suggested by the STOVL/Powered-Lift Technology Branch (FAP), NASA Ames Research Center.

Description

The basic purpose of this project was to produce an easy to use, and modify computer code which estimated lift increments due to power induced effects of V/STOL aircraft, in hover and forward flight, using methods developed by Kuhn and Stewart (References 3 and 6).

Philosophy

The PIE FORTRAN code is written in a modular structure which makes it easy to understand and modify. The code is divided into five fundamental sections; control, input, pre-calculations, lift increments, and output. Each of these modules are divided into individual parts to make logic and coding easier to understand.

Control

The Power Induced Effects (PIE) module is controlled by the control program, COPPIE, which coordinates the execution of PIE. COPPIE's first task is to make calls to input and pre-calculation subroutines. These calls are made first so all variables are either defined by the user or assigned a default value. Within these variables the number, limits, and steps are defined for each independent variable; height, velocity, nozzle deflection angle, and angle of attack. COPPIE steps from one to the maximum number of variations for each independent variable. (i.e. If 15 heights are to be considered then COPPIE steps from one to 15 for height.) The actual value for the independent variable is calculated and then the control routine for a lift increment(s) is called and/or the next independent variable is incremented. After all independent variables have been incremented up to their maximum values then the output subroutine is called. Figure #1 is a flow chart which shows the basic structure of COPPIE.

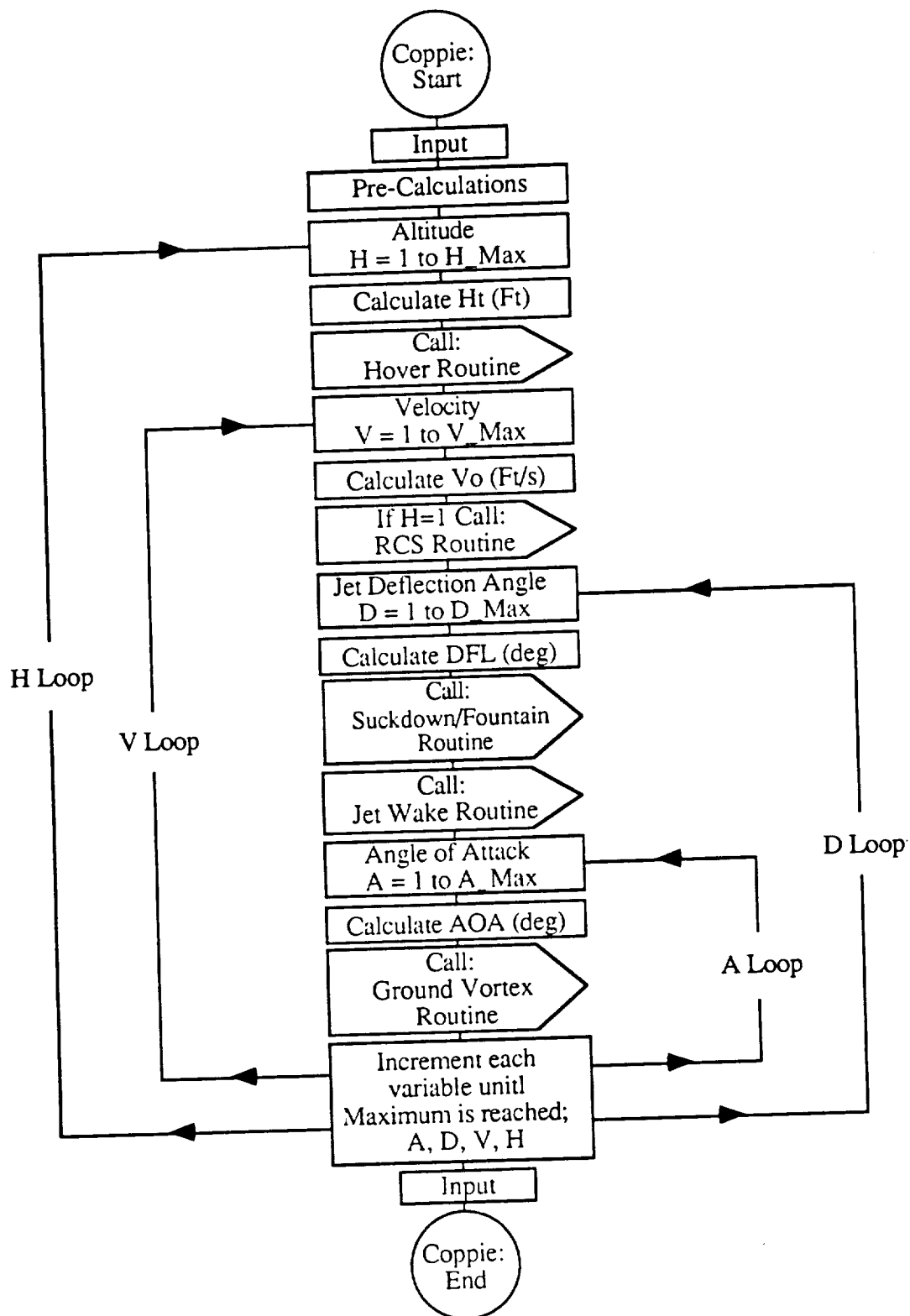


Figure #1 Flow chart of the control program COPPIE. (Note - Flow is downward unless specified.)

Each lift increment routine has its own control routine. This routine checks all configuration variables and passes only those variables which the lift increment routines require. This control routine isolates configuration variables from changes made inside lift increment routines and makes the code easier to understand. Further descriptions of these routines are provided in the Lift Increment Routines section.

Input

Input routines for PIE perform two functions; input of variables, and check/modification of variables for consistency.

The first routine of the input routines reads two input files. The first file contains all user defined variables. The second file contains X and Y coordinates of half the configuration planform. This routine first sets default values for all variables and then reads user defined variables from the first file. The advantage of this style of input over the input in the original programs (variables are entered in the code, hard-coded, Reference 4) is it allows the user to input as many or as few variables (above the minimum configuration variables) as the user desires. It also allows the user to save the configuration in a small, easy to manipulate files and rerun particular configurations as desired. The planform file (config.pie) allows a majority of calculation-intensive variables to be calculated by PIE, which alleviates the user from making tedious calculations.

The second routine assures planforms defined in the second file (config.pie) are consistent with the coordinate system defined in PIE. If the nose of the configuration does not lie on the origin, the routines modify all coordinates of the planforms and necessary configuration variables so the nose does lie on the origin. This makes sure PIE is able to interpret user inputs correctly.

Preliminary Calculations

A desirable feature of the Power Induced Effects Module (PIE) is it contains a complete set of variables which defines the aircraft. This allows lift increment control routines to choose variables needed by lift increment routines. Many pre-calculations need to be made to obtain this complete set of variables. These pre-calculations take a number of hours by hand for each configuration and they change with each new configuration.

Some of these preliminary calculations were very extensive, such as calculations for fountain areas (S_{FA1} , SP_{FA2} , etc...) and \bar{D} , (DB_B , DB_W , etc...). Many other preliminary calculations were very simple calculations such as nozzle areas (S_F , S_{FR}) and nozzle distances (X_{FR} , Y_R , etc...) but the number of these simple calculations is large.

Lift Increments Routines

The lift increment routines were divided into four sections. The sections are based on which independent variables are used to calculate the lift increment. The first section is lift increments calculated for hover. These lift increments vary with height. The second section contains lift increments which vary with velocity. The third section contains lift increments which vary with height, velocity, and jet deflection angle. The fourth section contains lift increments which change with height, velocity, jet deflection angle, and angle of attack.

Height

The hover routines calculate initial lift increments for the suckdown, and fountain which vary with height. They are later adjusted for forward velocity and jet deflection angle. The RCS, ground vortex, and jet wake lift increments occur when a freestream is present so they are not included in these calculations. The out-of-ground effect (OGE) suckdown is calculated first and then corrections are made for in-ground effect (GE) based on the altitude of the configuration. The fountain terms are calculated based on their proximity to the ground using the "Basic" and "h'" methods outlined by Kuhn (Reference 6).

The specific function of the hover control routine is to recognize when a high wing is present and make appropriate adjustment to variables which it sends to the lift increment routine.

Forward Velocity

The reaction control system (RCS) lift increment varies with velocity. Ground effects can be neglected since the size of the RCS jets are much smaller and their height above the ground, relative to their diameter, is much greater than the main lifting jets. Corrections for jet placement are calculated first. The lift increment based on pressure ratio is calculated next and adjustment factors are applied to the results.

The specific function of the RCS control routine is to either execute the lift increment routine or set its value to zero. This is controlled with a flag variable which can be changed by the user.

Height, Forward Velocity, and Jet Deflection Angle

Suckdown, fountain, and jet wake lift increments vary with height, forward velocity and jet deflection angle. Suckdown and fountain terms are calculated inside the same lift increment routine because they both change similarly with the independent variables. A height factor is calculated based on the forward extent of the wall jet. Then the calculations use hover values multiplied by height, forward flight and nozzle deflection angle factors. The control routine for suckdown and fountain routine track the positions of the nozzles and whether there is a high wing and passes the correct information to the suckdown and fountain lift increment routine.

The jet wake terms are calculated using methods described by Stewart (Reference 13). Since the methods makes separate calculations for different planforms and for front and rear jets, the control program is much more detailed than the others. The control program first determines which kind of planforms are to be used; wing and body or wing/body (one planform which is a combination of the wing and body.) It checks if the nozzles are near the trailing edge of the wing, for the jet flap effect, and then calculates the OGE and overall lift increment for the jet wake from front and rear nozzles. This process is repeated for the next planform. The OGE and overall lift increments are calculated in separate routines because of the complexity of each calculation. The calculations for the wing and body are stored separately to allow contributions of each planform to be observed.

Height, Forward Velocity, Jet Deflection Angle, and Angle of Attack

The ground vortex term is calculated based on height, forward velocity, nozzle deflection angle, and angle of attack. This is the only term which varies with angle of attack.

The control routine for the ground vortex term is similar to the jet wake control routine with respect to the planforms. The rear nozzles do not contribute to the ground vortex so only the front nozzles are used in the calculations. Lift increments for the wing and body are calculated and stored separately to allow contributions of each planform to be observed

Output

Storage of the results of PIE are in files and arrays which can be recalled with the use of an index notation. This is done so any type of lookup table can be created by the user/programmer. This method of storage allows the programmer to configure an output table in any format using simple index notation when requesting a value. (i.e. If the user wants a lift increment for the eighth height, sixth velocity, third deflection angle and fifth angle of attack, it can be recalled by setting the appropriate index variables; i, j, k, and l to 8, 6, 3, and 5 respectively.) This feature allows easy access to the lift increments which facilitates many different types of lookup table output.

Currently, PIE creates one three-dimensional lookup table for ACSYNT. This table contains multiple two-dimensional tables based on deflection angle with each individual table based on height and velocity. This table is generated by a single, short subroutine which demonstrates how many other types of tables can be generated using similar subroutines.

Other output options are created for viewing purposes. These tables are examples of the many tables which can be generated. Each lift increment as well as the total lift increments can be viewed based on two of their four independent variables. Currently, outputs are text files which can be read by a plotting routine, developed at NASA Ames for the Silicon Graphics IRIS workstations, and a Macintosh graphing application called KaleidaGraph™.

Limitations

PIE cannot calculate lift increments for configurations with five or more jets or configurations with three or more jets in a straight line either laterally or longitudinally. A temporary solution to this problem is to combine closely spaced jets into single jets, which can be rectangular or oval, until the total number of jets is reduced to four or less.

The methods used in this code are intended for use in preliminary design analysis and to give an indication of the effects of changes to primary configuration variables. Power induced effects are a complex function of many configuration variables and development of a V/STOL aircraft will require careful experimental investigations to accurately determine the induced forces.

Variables

Definitions

Below are names, default values, units and descriptions for all variables which can be modified. They are divided into sections to make it easier to find a particular variable. These sections are Control Variables, Body Variables, Center Section Variables, Wing Variables, Wingbody Variables, Forward Nozzle Variables, Rear Nozzle Variables, Front/Rear Nozzle Variables, RCS Variables, Fountain Variables, and Miscellaneous Variables.

Most of the variable names were created to enable them to be easily recognized and understood. All variables consist of six characters or less. The majority of them consist of a logical description of the variable along with a description as to what the variable applies, divided by a "_". An example is the variable S_WB, which is the planform area of the wingbody. Some of the other variables were created based on the same variables from other programs (Kuhn, Reference 4).

The Default column contains values for the variables which have default values. Any variable which does not have a default value will contain either "Calculated" or a "." possibly followed by a letter. The variables with "Calculated" in the Default column signify the variable will be calculated based on other variables unless it is defined by the user. The variables with a "." only are the minimum necessary input variables, these need to be defined. The variables with "." followed by a letter are also minimum input variables except they have special considerations given in the Minimum Configuration Variables section. Variables with the same letters are related to one another. All variables which need to be calculated are initially set to a flag value which is 9999.0. This is the only value which a variable cannot be set. The only variables which need to be watched are thrust variables (T_F, T_R, T_FR) which can be 9999.0 for particular configurations.

The Type column contains a one character description of the type of variable which is used. The character "R" refers to a real value, "I" refers to an integer value, "T" refers to a character value, and "L" refers to a logical value.

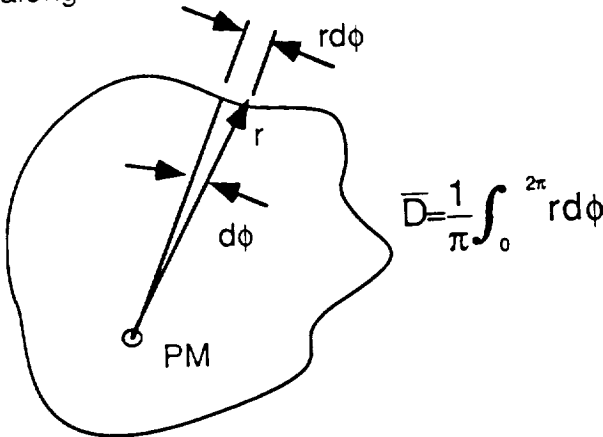
The Units column contains English units of the variable. The "----" indicates the variable is dimensionless.

The Description of Variable column contains a short description along with any special considerations needed when using the particular variable.

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
<u>Control Variables</u>				Contains all variables which deal with changing of the independent variables. The number of independent variables are restricted to the amount of memory available from the computer system. There are no error flags when a mismatch between the start, end, step and number variables. Therefore care must be taken to make sure they are consistent. It is possible to mismatch these variables and still have correct output but the start, end, or step values might not be the same as they were input. An example of this is shown in the velocity calculations (VSTEP).
AEND	30.0	R	deg	Ending value for angle of attack (AOA) - AOA is the angle between the nose of the aircraft and free stream (nose up is positive). This value is typically not greater than 30° to 45°. It could also be less than ASTART. If this variable is set to 9999.0 and ASTART, ASTEP and LA are defined then AEND will be calculated.
ASTART	0.0	R	deg	Starting value for angle of attack - See AEND for definition of AOA. This value is typically around 0° but it could be greater than AEND. If this variable is set to 9999.0 and AEND, ASTEP, and LA are defined then ASTART will be calculated.
ASTEP	Calculated	R	deg	Step value for angle of attack - See AEND for definition of AOA. When defining this variable, just to be safe, make sure steps are equally divisible by the difference between AEND and ASTART. This value is already defaulted

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				to be calculated so no modification is needed.
LA	4	I	----	Total number of Angle of Attack values - This value is typically small (<10) because only lift increments which change with angle of attack are those for the ground vortex. It is the least important of the four independent variables. If this variable is set to 9999 and AEND, ASTART, and ASTEP are defined then LA will be calculated. It is possible to set this value to 1 and set AEND and ASTART equal so only one angle of attack is used.
DEND	30.0	R	deg	Ending value for jet deflection angle (DFL) - DFL is the angle between the nozzles and body of the aircraft (0° is pointing aft (forward flight), and 90° is pointing (down)). This value can be anywhere between 0° and 90°. Similar to AEND when calculating this variable.
DSTART	90.0	R	deg	Starting value for jet deflection angle - See DEND for definition of DFL. Similar to ASTART when calculating this variable.
DSTEP	Calculated	R	deg	Step value for jet deflection angle - See DEND for definition of DFL. Similar to ASTEP when calculating this variable.
LD	7	I	----	Total number of nozzle deflection angles. - See DEND for definition of DFL. Similar to LA when calculating this variable. This values is typically smaller compared to height and velocity independent variables.
HEND	20.*DE_FR	R	Ft	Ending value for altitude (HT) - HT is the altitude the bottom of the aircraft is above the surface of the ground. This value <u>must</u> be above HSTART. The default value was chosen because at altitudes higher than 20 effective diameters of the nozzle, the lift increments do not change significantly. Similar to AEND when calculating this variable.
HSTART	2.0*DE_FR	R	Ft	Starting value for altitude - See HEND for definition of HT. This variable must be below HEND. The default value was chosen because the inaccuracy of the

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				method becomes too great at heights below two effective diameters of the nozzles.
HSTEP	Calculated	R	Ft	Step value for altitude - See HEND for definition of HT. It is desirable to have small steps between HSTART and HEND so as to distinguish various changes in lift increments. Similar to ASTEP when calculating this variable.
LH	25	I	----	Total number of height values. - See HEND for definition of HT. Similar to LA when calculating this variable. This independent variable is one of the two most important independent variables.
VEND	100.0	R	kts	Ending value for aircraft velocity (VO) - VO is the forward velocity of the aircraft. This value need not be greater then VSTART. The default value was chosen because it was a typical velocity which an aircraft might achieve wingbourn flight. Similar to AEND when calculating this variable.
VSTART	1.0	R	kts	Starting value for aircraft velocity - See VEND for definition of VO. The value is typically low but could be higher than VEND. Similar to ASTART when calculating this variable.
VSTEP	Calculated	R	kts	Step value for aircraft velocity - See VEND for definition of VO. The calculated value will cause the ending value to be 101 which is different than the input for VEND. Similar to ASTEP when calculating this variable.
LV	20	I	----	Total number of velocity values - See VEND for definition of VO. Similar to LA when calculating this variable. This independent variable is one of the two most important independent variables.
<u>Body Variables</u>				Body variables are determined if a body configuration is used as an input in the configuration file (config.PIE). The body planform is always input with a wing planform and never input with a wingbody planform.
B_B	Calculated	R	Ft	Width of Body planform - B_B is defined as maximum thickness of the body

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				planform. If the body planform contains a horizontal tail or canards then this value must be defined in the input file.
DB_B	Calculated	R	Ft	<p>Dbar of Body planform - Angular mean diameter. This is the diameter of a circle which has the same perimeter length as the given planform calculated using the equation below. The diagram of a general planform is shown below. Dbar changes with placement of the point of measure. The point of measure (PM) for the body planform is the point equal distance between the front and rear nozzles or if there is only one nozzle the point of measure is at the front nozzles along the X-axis.</p>  $\bar{D} = \frac{1}{\pi} \int_0^{2\pi} r d\phi$
L_B	Calculated	R	Ft	Length of body planform - L_B is calculated by the maximum distance from the nose of the body planform.
N_BODY	Calculated	I	----	Number of data points for Body planform - PIE counts the number of points used in the body planform definition. There is no reason to input this variable.
R_B	9999.	R	Ft	Corner radius of body sides - R_B is used in calculating the body contour factor (KR). It is defined as the radius of curvature of the sides of the body
S_B	Calculated	R	Ft ²	Area of Body - S_B is the total planform area of body. Calculated from planform input in the *.pie file.
WL_B	Calculated	R	----	Width to Length ratio of Body - WL_B is used in calculating lift increments for the ground vortex and the jet wake.

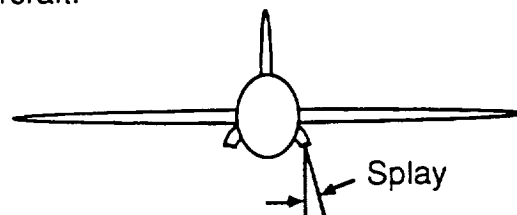
<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
X_BODY(500)	Calculated	R	Ft	X-coordinates of Body planform - Coordinate which defines X-coordinate of body planform. Coordinates are to start at the nose (0.0, not necessary) and progress toward the tail with increasing coordinate values. These values are obtained from the *.pie file.
Y_BODY(500)	Calculated	R	Ft	Y-coordinates of Body planform - Coordinate which defines Y-coordinate of body planform. Coordinates are to start at the center line of the body and increase in value toward the right (as seen by the pilot). These values are obtained from the *.pie file.
Z_B	0.0	R	Ft	Height of body base above nozzle - This value can be either positive or negative. A negative value denoting a body base below the nozzle. This variable is used in calculating lift increments for the ground vortex and the jet wake.
<u>Wing Variables</u>				Wing variables are determined if a wing configuration is used as an input in the configuration file (*.PIE). The wing planform is always input with a body planform and never input with a wingbody planform.
B_W	Calculated	R	Ft	Width of Wing (Wing span) - B_W is defined as the maximum thickness of the wing planform.
DB_W	Calculated	R	Ft	Dbar of Wing - Angular mean diameter of wing planform. See DB_B for a more detailed description.
MAC	.	R	Ft	Mean Aerodynamic Chord of wing
N_WING	Calculated	I	----	Number of data points for Wing planform - PIE counts the number of points used in the wing planform definition. There should be no reason to input this variable.
S_W	Calculated	R	Ft ²	Area of Wing - S_W is the total planform area of wing. Calculated from planform input in the *.pie file.
X_WING(500)	Calculated	R	Ft	X-coordinates of Wing planform - Coordinate which defines the X-coordinate of wing planform. The coordinates are to be placed on the same coordinate axis as the body and progress

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				toward the tail with increasing coordinate values. These values are obtained from the *.pie file.
Y_WING(500)	Calculated	R	Ft	Y-coordinates of Wing planform - Coordinate which defines Y-coordinates of the wing planform. The coordinates for the leading edge of the wing are to start at the center line of the wing and have a positive value toward the right (as seen by the pilot). These values are obtained from *.pie file.
Z_W	0.0	R	Ft	Height of wing above nozzle - This value can be either positive or negative. A negative value denoting a wing below the nozzle (unusual). This variable is used in calculating lift increments for hover, ground vortex and jet wake.
<u>Wingbody Variables</u>				Wingbody variables are determined if a wingbody configuration is used as an input in the configuration file (*.PIE). This configuration is always input alone without the body or wing planform. It is used for single body models such as flat-plate models. Whenever a wing and body are input then the wing-body planform is created from a combination of the wing and body planforms.
B_WB	Calculated	R	Ft	Width of Wing-Body - B_WB is defined as the maximum thickness of the wingbody planform.
DB_WB	Calculated	R	Ft	Dbar of Wing-Body - Angular mean diameter of wing planform. See DB_B for a more detailed description.
L_WB	Calculated	R	Ft	Length of Wing-Body - L_WB is calculated by the maximum distance from the nose of the wingbody planform.
N_WB	Calculated	I	---	Number of data points for Wing-Body planform - PIE counts the number of points used in the wing-body planform definition. There should be no reason to input this variable.
S_WB	Calculated	R	Ft ²	Area of Wing-Body planform - S_WB is the total planform area of wing-body. Calculated from planform input in the *.pie file or from planform made from wing and body planform.

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
WL_WB	Calculated	R	----	Width to Length ratio of Wing-Body - WL_WB is used in calculating lift increments for the ground vortex and the jet wake.
X_WB(500)	Calculated	R	Ft	X-coordinates of Wing-Body planform - Coordinate which defines X-coordinates of wing-body planform. The coordinates are to be placed on the same coordinate axis as the body and progress toward the tail with increasing coordinate values. These values are obtained from *.pie file or determined by PIE given the wing and body planforms.
Y_WB(500)	Calculated	R	Ft	Y-coordinates of Wing-Body planform - Coordinate which defines the Y-coordinate of the wing-body planform. The coordinates are to start at the center line of the wing-body and have a positive value toward the right (as seen by the pilot). These values are obtained from the *.pie file or determined by PIE given the wing and body planforms.
<u>Center Section Variables</u>				Center section variables are determined if a wing and body configuration are used as an input in the configuration file (config.PIE). The center section variables are created from a combination of the wing and body planforms.
B_CS	Calculated	R	Ft	Width of Center Section - B_CS is defined as the maximum thickness of the wingbody planform. Calculated when wing and body planforms are input. Used in calculating lift increments for the jet wake.
B_JP	Calculated	R	Ft	Width of Jet Pattern - Width of pattern defined by the placements of the jets (each jet is a corner). Used in calculating WL_JP.
DB_CS	Calculated	R	Ft	Dbar of Center Section - Angular mean diameter of center section. See DB_B for a more detailed description.
L_CS	Calculated	R	Ft	Length of Center Section - L_CS is calculated by the maximum distance from the front to the rear of the center section. Used in calculating WL_JP.

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
PP_LID	•E	R	----	Ratio of perimeter enclosed by lift improvement devices (LIDS) to total perimeter of LIDS. See Kuhn (Ref 1)
S_CS	Calculated	R	Ft ²	Area of Center Section - S_CS is the total planform area of the center section. Calculated when a wing and body planform are input. Used in calculating lift increments for the ground vortex and jet wake.
S_JP	Calculated	R	Ft ²	Area enclosed by Jet Pattern - S_JP is the total planform area inside the jet pattern defined by the nozzles. Used in calculating lift increments for the hover fountain.
S_LID	•E	R	Ft ²	Area enclosed by LIDS - S_LID is the total planform area inside the lift improvements devices. Used in calculating lift increments for the fountain.
SP_JP	•E	R	Ft ²	Actual surface area within area enclosed by nozzles - SP_JP can be less than or equal than S_JP but never greater. Used in calculating lift increments for the fountain core and LIDS.
WL_CS	Calculated	R	----	Width to Length ratio of Center Section - Not used in any current calculations
WL_JP	Calculated	R	----	Width to Length ratio of Jet Pattern - WL_JP is used in calculating the lift increments for the hover fountain.
<u>Forward Nozzle Variables</u>				Forward nozzle variables are always defined. If there is only one set of nozzles, they will be defined at the forward nozzles. Variables for the nozzles are input using the *.dat files.
D_F	•B	R	Ft	Diameter of each Front jet - D_F is the diameter of an individual front circular or oval nozzle. If there are two nozzles then input only the diameter of one of them. If the nozzle is oval then input the diameter which would give a circle the same area as the oval. If the nozzle is rectangular then only input the length and width.
DE_F	Calculated	R	Ft	Effective Diameter of Front jets - DE_F is the effective diameter of the front jets combined. This is the diameter of a single circle which would give the same area as the total area of the front nozzles.

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
L_F	•B	R	Ft	Length of Front jet - L_F is the length in the Y-direction of the front jet(s). Input this and W_F only if the nozzles are rectangular, not D_F.
NUM_F	Calculated	I	----	NUMber of Front jets - NUM_F is the total number of front jets present. This number cannot be greater than two. This is calculated by examining the location of the jets.
PR_F	•C	R	----	Jet Pressure Ratio for front jets - PR_F used to calculate the lift increments for the hover suckdown and the jet wake. Also used to calculate the dynamic pressure of the front nozzles.
Q_F	Calculated	R	lb/Ft ²	Dynamic pressure for the front jets - Q_F is used to calculate the lift increments for the jet wake.
S_F	Calculated	R	Ft ²	Area of Front jets - Total area of front jets
SF_FB	Calculated	R	Ft ²	Area ahead of Front jets using body planform - Total area of body planform ahead of the front jets. Used to calculate the lift increments for the jet wake.
SF_FW	Calculated	R	Ft ²	Area ahead of Front jets using wing planform - Total area of wing planform ahead of the front jets. Used to calculate the lift increments for the jet wake.
SF_FWB	Calculated	R	Ft ²	Area ahead of Front jets using the wingbody planform - Total area of wing-body planform ahead of the front jets. Used to calculate the lift increments for the jet wake.
SPLY_F	0.0	R	Ft	SPLaY angle of Front jet - SPLY_F is the angle from the vertical which the nozzles are canted. A positive splay angle is one which is away from the center of the aircraft.



T_F	•A	R	lb	Thrust of Front jets - Total thrust of all front jets. (Make sure this value is not 9999.0)
-----	----	---	----	---

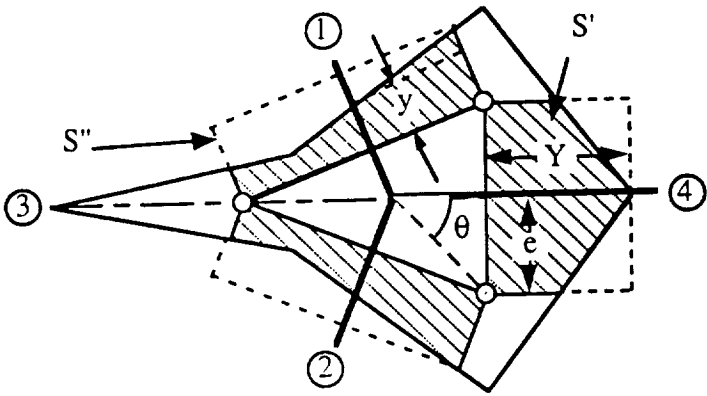
<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
TT_F	•A	R	----	Front Thrust / total Thrust
W_F	•B	R	Ft	Width of Front jet - W_F is the width in the X-direction of the front jet(s). Input this and L_F only if the nozzles are rectangular, not D_F.
WL_F	Calculated	R	----	Width to Length ratio of Front jets
XCG_F	Calculated	R	Ft	Distance Front jet is ahead of CG - XCG_F is the distance between the center of gravity (CG) and the front nozzles. A positive value denotes that the nozzles are ahead of the CG.
XNOZ_F	•	R	Ft	X-coordinates of Front NOZZle - Position of the front nozzles along the X-axis.
XTE_F	•	R	Ft	X-coordinate of trailing edge for front nozzle - Position along the X-axis of the wing trailing edge (TE) directly behind or in front of the front nozzle(s). Use wing root TE if Nozzle is within the body planform. (This value can be different from the XTE_R). Use in calculating the lift increment due to the jet flap condition for the jet wake. (This is not the distance between the front nozzle and the trailing edge, but the X-coordinate of the trailing edge behind the front nozzle)
Y_F	Calculated	R	Ft	Distance between Front jets - Lateral distance (Y-direction) between the front nozzles.
YB_F	Calculated	R	Ft	Lateral distance from Body to Front jets - YB_F is the lateral distance (Y-direction) from the side of the body to the nozzles. An external nozzle will have a positive YB_F.
YNOZ_F	•	R	Ft	Y-coordinates of Front NOZZle - Position of the front nozzles along the Y-axis. Only one value for the front nozzles need to be entered due to the symmetry of the configurations. If YNOZ_F is 0.0 then only one nozzle is assumed. If YNOZ_F is not equal to zero then, due to symmetry, two nozzles are assumed.
YWB_F	Calculated	R	Ft	Lateral distance from Wingbody to Front jets - YWB_F is the lateral distance (Y-direction) from the side of the wing-body to the nozzles. An external nozzle will have a positive YWB_F.

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
<u>Rear Nozzle Variables</u>				Rear nozzle variables are defined only if there are front and rear nozzles. The variables for the nozzles are input using the *.dat files.
D_R	•B	R	Ft	Diameter of each Rear jet - D_R is the diameter of an individual rear circular or oval nozzle. If there are two nozzles then input only the diameter of one of them. If the nozzle is oval then input the diameter which would give a circle the same area as the oval. If the nozzle is rectangular then only input the length and width.
DE_R	Calculated	R	Ft	Effective Diameter of Rear jets - DE_R is the effective diameter of the rear jets combined. This is the diameter of a single circle which would give the same area as the total area of the rear nozzles.
L_R	•B	R	Ft	Length of Rear jet - L_R is the length in the Y-direction of the rear jet(s). Input this and W_R only if the nozzles are rectangular, not D_R.
NUM_R	Calculated	I	----	NUMBER of Rear jets - NUM_R is the total number of rear jets present. This number cannot be greater than two. This is calculated by examining the location of the jets.
PR_R	•C	R	----	Jet Pressure Ratio for rear jets - PR_R used to calculate the lift increments for the hover suckdown and the jet wake. Also used to calculate the dynamic pressure of the rear nozzles.
Q_R	Calculated	R	lb/Ft ²	Dynamic pressure for the rear jets - Q_R used to calculate the lift increments for the jet wake.
S_R	Calculated	R	Ft ²	Area of Rear jets - Total area of the rear jets.
SF_RB	Calculated	R	Ft ²	Area ahead of Rear jets using body planform - Total area of body planform ahead of the rear jets. Used to calculate the lift increments for the jet wake.
SF_RW	Calculated	R	Ft ²	Area ahead of Rear jets using wing planform - Total area of wing planform ahead of the rear jets. Used to calculate the lift increments for the jet wake.
SF_RWB	Calculated	R	Ft ²	Area ahead of Rear jets using the wingbody planform - Total area of wing-

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				body planform ahead of the rear jets. Used to calculate the lift increments for the jet wake.
SPLY_R	0.0	R	Ft	SPLaY angle of Rear jet - SPLY_R is the angle from the vertical which the nozzles are canted. A positive splay angle is one which is away from the center of the aircraft. See diagram for SPLY_F.
T_R	•A	R	lb	Thrust of Rear jets - Total thrust of all rear jets. (Make sure this value is not 9999.0)
TT_R	•A	R	----	Rear Thrust / total Thrust
W_R	•B	R	Ft	Width of Rear jet - W_R is the width in the X-direction of the rear jet(s). Input this and L_R only if the nozzles are rectangular, not D_R.
WL_R	Calculated	R	----	Width to Length ratio of Rear jets
XCG_R	Calculated	R	Ft	Distance Rear jet is ahead of CG - XCG_R is the distance between the center of gravity (CG) and the rear nozzles. A negative value denotes that the nozzles are behind the CG.
XNOZ_R	•	R	Ft	X-coordinates of Rear NOZzle - Position of the rear nozzles along the X-axis.
XTE_R	•	R	Ft	X-coordinate of trailing edge for rear nozzle - Position along the X-axis of the wing trailing edge (TE) directly behind or in front of the rear nozzle(s). Use wing root TE if Nozzle is within the body planform. (This value can be different from the XTE_F). Use in calculating the lift increment due to the jet flap condition for the jet wake. (This is not the distance between the rear nozzle and the trailing edge, but the X-coordinate of the trailing edge behind or in front of the rear nozzle)
Y_R	Calculated	R	Ft	Distance between Rear jets - Lateral distance (Y-direction) between the rear nozzles.
YB_R	Calculated	R	Ft	Lateral distance from Body to Rear jets - YB_R is the lateral distance (Y-direction) from the side of the body to the nozzles. An external nozzle will have a positive YB_R.
YNOZ_R	•	R	Ft	Y-coordinates of Rear NOZzle - Position of the rear nozzles along the Y-axis. Only one value for the rear nozzles need to be

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				entered due to the symmetry of the configurations. If YNOZ_R is 0.0 then only one nozzle is assumed. If YNOZ_R is not equal to zero then, due to symmetry, two nozzles are assumed.
YWB_R	Calculated	R	Ft	Lateral distance from Wingbody to Rear jets - YWB_R is the lateral distance (Y-direction) from the side of the wing-body to the nozzles. An external nozzle will have a positive YWB_R.
<u>Front/Rear Nozzle Variables</u>				Front/rear nozzle variables are always calculated even if there are only front nozzles. The user should only need to input certain front/rear nozzle variables
DE_FR	Calculated	R	Ft	Effective Diameter of Front and Rear jets combined - DE_F R is the effective diameter of the all the jets combined. This is the diameter of a single circle which would give the same area as the total area of the front and nozzles combined.
NUM	Calculated	I	Ft	NUMBER of Front/Rear jets - NUM is the total number of all jets present. This number cannot be greater than four. This is calculated summing NUM_F and NUM_R.
PER_FR	Calculated	R	Ft	Total perimeter of jets - PER_FR is the total perimeter of all the jets combined. Calculation of this takes into account whether the jets are circular, oval, or rectangular.
PR_FR	•C	R	----	Jet Pressure Ratio for all jets - PR_FR used to calculate the lift increments for the hover suckdown and the jet wake. Also used to calculate the average dynamic pressure of all the nozzles.
Q_FR	Calculated	R	lb/Ft ²	Dynamic pressure for the front/rear jets - Q_FR is used to calculate the lift increments for the jet wake.
S_FR	Calculated	R	Ft ²	Area of Front/Rear jets - Total area of front jets
T_FR	•A	R	lb	Thrust of Front/Rear jets - Total thrust of all jets. (Make sure this value is not 9999.0)
X_FR	Calculated	R	Ft	Distance between Front and Rear jets - X_FR is the longitudinal distance

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				between the front and rear nozzle locations.
<u>RCS Variables</u>				Typically these variables are only input when moment increments are being calculated. Since the PIE code does not contain moment increments at this time these variables need only be input when the user is interested in the lift increments caused by the RCS.
D_RCS	•D	R	Ft	Diameter for Roll RCS nozzle
MD_RCS	•D	R	lb _m /s	Mass flow rate for one RCS nozzle
PR_RCS	•D	R	----	Roll RCS Pressure Ratio
PT_RCS	•D	R	lb/Ft ²	Total pressure for one roll RCS jet
Q_RCS	•D	R	lb/Ft ²	Dynamic pressure for roll RCS jets
RCSFLG	Calculated	R	----	Flag which identifies if the RCS lift increment calculation exceeded the area ratio. This is not a fatal error. $\left(\frac{S_{wing}}{A_{jet}} > 7000\right)$ TRUE - Exceeded FALSE - Not Exceeded
T_RCS	•D	R	lb	Thrust of one roll RCS nozzle
TP_RCS	•D	R	°R	Temperature of flow in roll RCS nozzle
X_RCS	•D	R	Ft	Distance of roll RCS nozzle ahead of wing trailing edge
Y_RCS	•D	R	Ft	Distance of roll RCS nozzle from wing tip
<u>Fountain Variables</u>				All of these variables, except for SCALE3, are calculated by PIE. They should typically not be changed unless the user wants to compare the PIE results with other results which have different values than calculated by PIE. Setting these variable to user defined values will not speed up the calculations because the SDAREA is still executed. Refer to the diagram for the fountain variables. The fountain arms are numbered as shown in the figure. There can be a maximum of four fountain arms (configurations with four or more jets.) The number two fountain arm is a reflection of the number one fountain arm so it is not included in the variable listing. The figure shown has only three fountain

Name	Default	Type	Units	Description of Variable
				arm so only fountain arms number one and four need to be defined.
				
E_1	Calculated	R	Ft	Half distance between adjacent jets for fountain arm (1). ('e' in above figure)
E_3	Calculated	R	Ft	See E_1 - use fountain arm (3).
E_4	Calculated	R	Ft	See E_1 - use fountain arm (4).
S_FA1	Calculated	R	Ft ²	Actual surface area between the jets which is affected by fountain arm (1). (S' in above figure)
S_FA3	Calculated	R	Ft ²	See S_FA1 - use fountain arm (3).
S_FA4	Calculated	R	Ft ²	See S_FA1 - use fountain arm (4).
SCALE	Calculated	R	----	Actual scale factor which adjust the area of fountain arm (3) to account for the fact that the curvature of the aircraft's nose tends not to hold the fountain arm causing the area that the fountain arm affects to be scaled down.
SCALE3	1.0	R	----	Fountain arm (3) scalar (Percentage measured from the front jets to the nose of the aircraft.) Tries to approximate SCALE but does not do a great job (better than nothing.)
SP_FA1	Calculated	R	Ft ²	Potential surface area between the jets which is affected by fountain arm (1). (S'' in above figure)
SP_FA3	Calculated	R	Ft ²	See SP_FA1 - use fountain arm (3).
SP_FA4	Calculated	R	Ft ²	See SP_FA1 - use fountain arm (4).
THA_1	Calculated	R	deg	Half the angle between two adjacent jets and the fountain core which is bisected by fountain arm (1) ('theta' in above figure)
THA_3	Calculated	R	deg	See THA_1 - use fountain arm (3)
THA_4	Calculated	R	deg	See THA_1 - use fountain arm (4)

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
Y_1	Calculated	R	Ft	Spanwise extent of planform on fountain arm (1) center line. ('y' in above figure)
Y_3	Calculated	R	Ft	See Y_1 - use fountain arm (3)
Y_4	Calculated	R	Ft	See Y_1 - use fountain arm
YP_1	Calculated	R	Ft	Maximum spanwise extent of planform along fountain arm (1). ('Y' in above figure)
YP_3	Calculated	R	Ft	See YP_1 - use fountain arm (3)
YP_4	Calculated	R	Ft	See YP_1 - use fountain arm (4)
<u>Miscellaneous Variables</u>				
CONFIG	??????...	T	----	Short title of current configuration (18 characters or less)
DR	.Z	R	----	Density Ratio (Jet / Atm)
F_NAME	test.pie	T	----	Name of file which contains body & wing, or just wingbody planform coordinates. It should always end with '.pie' to distinguish it from the data file (Maximum length - 50 characters) F_NAME can contain a directory path to the '.pie' file.
HDEOUT	.TRUE.	L	----	Signals when to output tables based on height or height/De TRUE - Print table based on Height/DE, FALSE - Print table based on height
KB	.666667	R	----	Boundary layer factor. Used when calculating forward extent of wall jet. Value is based on whether the ground plane is moving ??
KLF	1.0	R	----	Adjustment factor for flap extension when calculating the lift increment due to jet induced effects. 1.0 - Flaps extended .25 - No flap extension
KR	Calculated	R	----	Body contour factor Calculated from R_B.
NDIV	500	I	----	Number of divisions to be used when calculating suckdown areas (SDAREA) and Dbars (DIABAR). This value has not been optimized. Value could be less and still have good accuracy.
PRTFLG	.TRUE.	L	----	Signals when to output to the screen. Prints all input variables according to logical section (similar to the sections in this manual) and then allows the user to choose which increments to create a

<u>Name</u>	<u>Default</u>	<u>Type</u>	<u>Units</u>	<u>Description of Variable</u>
				table from. PIE uses a menu driven structure for this purpose. Output file is written over the previous output each time a new table is created. (To create multiple tables, create table and then rename it before creating next table.)
VEOUT	.FALSE.	L	----	Signals when to output tables based on VE TRUE - Print table based on VE FALSE - Print table based on Velocity
WIWE	•Z	R	----	Flow Weight of Engine Inlet / Flow Weight of Engine Exits
X_CA	Calculated	R	Ft	X-coordinate of Center of Area measured from the nose of the aircraft.
X_CG	•	R	Ft	X-coordinate of Center of Gravity measured from the nose of the aircraft.
XCA_CG	Calculated	R	Ft	Distance of center of area ahead of CG (if value is negative, center of area is aft of the CG.)
XCG_C2	Calculated	R	Ft	Distance from CG to half chord (MAC/2) Not used in lift increment calculations.
XCG_C4	•Z	R	Ft	Distance from CG to quarter chord (MAC/4) Not used in lift increment calculations.
XCG_I	•Z	R	Ft	Engine Inlet longitudinal distance ahead of CG
ZCG_I	•Z	R	Ft	Engine Inlet vertical distance above CG

Minimum Configuration Variables

There are a certain number of variables which must be defined before PIE makes calculations. These variables are called minimum configuration variables. These variables are divided into seven groups; Necessary Group, Thrust Group, Nozzle Definition Group, Pressure Ratio Group, RCS Group, LIDs Group, and Potential Group. The variables in each group, except the Necessary Group, are related which is explained in the following sections. The headings for each section are preceded by a letter which corresponds to the letter found after the "•" in the Variable Definition section.

These variables are general variables. The engineer should read through the Variable Definition section to become familiar with the many different subtle configuration changes which might apply to an aircraft design. This section will list variable names and maybe some pertinent description. The complete definition of the variable can be found in the Variable Definition section.

Here are some points which the user should be aware of:

- 1) All variables must be entered using the same coordinate system.
- 2) If there are no rear nozzles then the user need not enter any variables which are specifically for the rear nozzles.

Necessary Group

The Necessary Group contains all those variables which must be entered in the input file. These variables have no conditions on them and therefore do not belong in one of the lettered groups.

X.CG

XNOZ_F

YNOZ_F

XNOZ_R Required if rear nozzles are present.

YNOZ_R Required if rear nozzles are present.

XTE_F

XTE_R Required if rear nozzles are present & value is different from XTE_F.

MAC

F_NAME

Thrust Group

The Thrust group contains all the variables pertaining to the thrust of the nozzles. Any one of the following groups of two can be entered. Do not enter more than two thrust variables. The only exception to this rule is if there are only front nozzles then it is only necessary to enter T_F or T_FR. If a different group of two, other than the ones listed here, is input then the thrust error flag, T_ERR, will be set to one. This error will not cause PIE to abort but it will cause incorrect results and therefore must be corrected.

T_F, T_R

TT_F, T_FR

TT_R, T_FR

TT_F, T_F

TT_R, T_R

T_FR, T_R

T_FR, T_F

Nozzle Definition Group

The Nozzle Definition Group contains the variables which define the size and type of nozzles. There are three different types of nozzles which PIE can handle; circular, rectangular and oval. If there are front and rear nozzles, each set must be defined as a type of nozzle using the following definitions

Circular nozzles are defined by entering the diameter of an individual nozzle in front or rear, if necessary. (D_F, D_R)

Rectangular nozzles are defined by entering the width and length of an individual nozzle in front or rear, if necessary. (W_F, L_F, W_R, L_R)

Oval nozzles are defined by entering the width and length of an individual nozzle along with the diameter of a circular individual nozzle which has the same area as the individual oval nozzle. Use rear nozzles only if necessary. (W_F, L_F, D_F, W_R, L_R, D_R)

Pressure Ratio Group

The Pressure Ratio Group contains all variables which define the pressure ratios of the nozzles. The preferable method for entering the nozzle pressure ratios is to enter the pressure ratios for each nozzle (PR_F, PR_R). If these are not known then the pressure ratio for the whole system must be entered (PR_FR).

RCS Group

The reaction control system group contains all variables which define the nozzles of the RCS. The following is a list of variables which are needed by PIE to compute RCS lift increments. If one of the variables cannot be determined and there is an 'or' on the same line then the next variable can be entered in its place. e.g. If the diameter of the RCS nozzle (D_RCS) cannot be found then the three values following it must all be used (MD_RCS, TP_RCS, PT_RCS).

If either X_RCS or Y_RCS are not defined then the RCS routine will not be executed.

X_RCS

Y_RCS

T_RCS

Q_RCS or PT_RCS or PR_RCS

D_RCS or MD_RCS
TP_RCS
PT_RCS

LIDs Group

The LIDs group contains all the variables which define the LIDs and how they are positioned on the airframe. These variables are to be included if the configuration contains LIDS

S_LID

PP_LID

SP_JP Required if different than S_JP.

Potential Group

The Potential Group of variables are variables which serve no purpose at this time. They are not needed within any calculations as of this writing. They are defined and positioned throughout PIE to allow the easy incorporation of the moment increment routines. These variables generally refer to moment arms which the forces will act upon to cause the moment increments. Since the lift increments do not need these variables, they can be left out of the input files until the moment increment routines are included.

DR

WIWE

XCG_C4

XCG_I

ZCG_I

Output

The storage of the results of PIE are in arrays. This is done so any type of lookup table can be created by a user/programmer. This method of storage allows the programmer to configure an output table in any form using simple index notation when requesting increments. This feature allows many different types of lookup table output.

Currently, PIE produces one three-dimensional lookup table, given an angle of attack, for ACSYNT. This table contains multiple tables based on deflection angle with each individual table based on height and velocity. This table is generated by a single, short subroutine which shows how other types of tables can be generated using similar subroutines.

The other output options are created for graphing purposes. These outputs are an example of the many tables which can be generated. Each lift increment, as well as, the total lift increments can be viewed based on two of four independent variables. Currently, outputs are text files which can be read by a plotting routine, developed at NASA Ames for the Silicon Graphics IRIS workstations, and a Macintosh graphing application called KaleidaGraph™.

Example

Configuration

The example configuration is a McDonnell Douglas YAV-8B, "Harrier" V/STOL aircraft which was originally developed by the British. This configuration was chosen because it is the most common jet lift V/STOL aircraft. One addition this example has over common configurations are LIDs.

Input

Input files can be created in any text editor available to the user. Make sure no special control characters are included in any of the text files.

The data file "config.dat" contains all user defined variables. This file contains namelist formatted input. This means the title of the namelist must be included first in the second column (indent one space) of the file. Following this namelist title on the second line the user defined variables can be entered in any order. The variables in the example file, Table #1, have been listed according to the groups listed above for convenience of the user. After these groups, the variables are positioned in no particular order. It is recommended variables be grouped together in logical groups to allow ease of identification of variables. A suggested group structure would be the groups found in the Variables Definition section of this manual. Each variable must be separated by a comma and one or more of the following; space, tab, or return.

Table #1. Sample Data File - YAV8B.dat

```
$PIENAM
  X_CG = 25.8,
  XNOZ_F = 25.3,   YNOZ_F = 3.39,
  XNOZ_R = 30.38,  YNOZ_R = 2.8,
  XTE_F = 29.348,  MAC = 8.31,
  T_F = 9994.,     T_R = 11423.,
  D_F = 1.7692,    D_R = 2.17,
  PR_FR = 2.25,
  PP_LID = 1.0,     S_LID = 18.27,
  SPL_Y_F = -10.,   SPL_Y_R = -10.,
  CONFIG = 'YAV-8B Harrier',   WL_F = .74,
  B_B = 6.36, KR = .6,
  XCG_C2 = -3.6,
  Z_W = 4.2,
  F_NAME = 'yav8b.pie',
  HDEOUT = .FALSE.
  HSTART = 1,       HEND = 25,       LH = 25.,
  VSTART = 3.0,     VSTEP = 3.5,     LV = 10.,
```

```

        ASTART = 6.0,    ASTEP = 2.0,    AEND = 12.0,
        DSTART = 81.,    DSTEP = 2.0,    DEND = 93.,
$END

```

The planform coordinates file, "config.pie", contains Cartesian coordinates of planforms used for the configuration. This file contains listings of coordinates for particular planforms. Only half the planform must be entered due to symmetry of the configurations.

The format for the coordinates starts with a descriptor of the planform which can be one of the following; WINGPLAN, BODYPLAN, or WINGBODY. Following these descriptors the coordinates for the planform are entered from nose to tail of the configuration. The first and last coordinates must be positioned on the center line of the aircraft (Y-coordinates equal to zero.) Each planform listing is terminated with the planform termination flags, 9999.0, in both the X and Y-coordinate positions. After the planform termination flags are entered the next planform can be entered using the same formats. Enter no more than 500 points for each planform. These formats can be seen in Table #2.

The preferred method for entering planforms is to enter the WINGPLAN and BODYPLAN planforms together. If these planforms cannot be determined then enter only one planform for the WINGBODY. Do not enter all three planforms at once.

Table #2. Example Planform Coordinates File - YAV8B.pie

```

WINGPLAN
21.3, 0.0
31.5, 14.0
32.9, 14.6
35.44, 14.6
34.0, 8.4
34.0, 0.0
9999., 9999.
BODYPLAN
0.0, 0.0
7.97, 0.21
11.0, 1.4
15.9, 1.4
16.7, 3.23
25.2, 2.91
29.4, 3.18
30.8, 2.3
36.3, 2.0
41.4, 1.07
44.5, 5.0

```

45.7, 6.7
 47.4, 6.7
 46.2, 0.54
 49.55, 0.0
 9999., 9999.

Execution

The set up for execution of PIE requires the input file "config.dat" be assigned or linked to the FORTRAN file, unit 9 (fort.9 for Silicon Graphics Iris.) PIE can then be executed by typing "pie". Depending on the machine, the code will execute for a few moments and the variable lists will be displayed. Table #3 is a reproduction of the variable lists.

Table #3. Sample Output During Execution

YAV-8B Harrier Body	Wing	Wing-Body	Jet Pattern	Center Sec.
B_B=6.36	B_W=29.20	B_WB=29.20	B_JP=6.78	B_CS=6.36
L_B=49.55		L_WB=49.55	S_JP=31.45	L_CS=12.75
S_B=197.48	S_W=224.30	S_WB=357.53	S_LID=18.27	S_CS=64.24
DB_B=12.39	DB_W=15.76	DB_WB=19.43	SP_JP=31.45	DB_CS=8.72
WL_B=0.13	MAC=8.31	WL_WB=0.59	WL_JP=1.33	WL_CS=0.50
Z_B=0.00	Z_W=4.20		PP_LID=1.00	
KR=0.60				

Table 3. (Continued)

Front Nozzles	Rear Nozzles	Both Nozzles	Misc. Nozzles
NUM_F=2	NUM_R=2	NUM=4	PR_FR=2.25
D_F=1.77	D_R=2.17		DR=1.00
DE_F=2.50	DE_R=3.07	DE_FR=3.96	WIWE=0.00
S_F=4.92	S_R=7.40	S_FR=12.31	XCG_I=0.00
SPLY_F=-10.0	SPLY_R=-10.0	PER_FR=24.75	ZCG_I=0.00
W_F=1.77	W_R=2.17	Q_FR=1761.6	
L_F=1.77	L_R=2.17	X_FR=5.08	
WL_F=1.00	WL_R=1.00		
Y_F=6.78	Y_R=5.60		
YB_F=0.47	YB_R=0.24		
YWB_F=-2.10	YWB_R=-9.66		
PR_F=2.25	PR_R=2.25		
T_F=9994.0	T_R=11423.0		
TT_F=0.47	TT_R=0.53		
Q_F=1761.6	Q_R=1761.6		
XTE_F=29.35	XTE_R=29.35		
XCG_F=0.50	XCG_R=-4.58		
SF_FB=76.75	SF_RB=107.37		
SF_FWB=81.48	SF_RWB=172.68		
XNOZ_F=25.30	XNOZ_R=30.38		
YNOZ_F=3.39	YNOZ_R=2.80		
Fount. Arms	RCS Nozzles	Misc.	
E_1=2.56	D_RCS=0.00	XCG_C2=-3.60	
E_3=3.39	PR_RCS=0.00	XCG_C4=0.00	
E_4=2.80	T_RCS=0.00	XCA_CG=-4.83	
THA_1=39.37	TP_RCS=0.00	KB=0.67	
THA_3=57.25	PT_RCS=0.00	KLF=1.00	
THA_4=44.00	MD_RCS=0.00	X_CG=25.80	
S_FA1=0.27	Q_RCS=0.0	X_CA= 30.63	
S_FA3=76.97	X_RCS=9999.00		
S_FA4=68.71	Y_RCS=9999.00		
SP_FA1=1.29			
SP_FA3=171.53			
SP_FA4=107.35			
Y_1=0.00			
Y_3=25.30			
Y_4=19.17			
YP_1=0.25			
YP_3=25.30			
YP_4=19.17			
SCALE3=1.00			

Table 3. (Continued)

Body X	Y	Wing X	Y	Wingbody X	Y	Center Sec X	Y
0.00	0.00	21.30	0.00	0.00	0.00	21.30	0.00
7.97	0.21	31.50	14.00	7.97	0.21	23.47	2.98
11.00	1.40	32.90	14.60	11.00	1.40	25.20	2.91
15.90	1.40	35.44	14.60	15.90	1.40	29.40	3.18
16.70	3.23	34.00	8.40	16.70	3.23	30.80	2.30
25.20	2.91	34.05	0.00	23.47	2.98	34.04	2.12
29.40	3.18	31.50	14.00	34.05	0.00		
30.80	2.30	32.90	14.60				
36.30	2.00	35.44	14.60				
41.40	1.07	34.00	8.40				
44.50	5.00	34.04	2.12				
45.70	6.70	36.30	2.00				
47.40	6.70	41.40	1.07				
46.20	0.54	44.50	5.00				
49.55	0.00	45.70	6.70				
		47.40	6.70				
		46.20	0.54				
		49.55	0.00				

Error Codes

 DBAR error flag (DBERR) = 0
 SDAREA error flag (SDAERR) = 0
 Thrust input error (T_ERR) = 0

After the variables are output to the screen, a menu system will appear which allows the user to choose which type of output required. The following menus and choices are the exact output of a typical PIE run. The first choice is for a table with all hover results, the second choice is for a table with total lift increments, from all routines, based on height for a velocity of 34.5 knots, nozzle deflection angle of 81 degrees, and angle of attack of 6 degrees. The third choice is a table of the totals which can be used by ACSYNT for an angle of attack of 6 degrees. (The output for all these choices are shown following this paragraph.)

OUTPUT DATA TO TABLE FORMAT

- 0 - Quit table generation section
- 1 - Output table with all hover results (dL/dT vs Height)
- 2 - Tables with results from STOLSF (any)
- 3 - Tables with results from STOLGV (any)
- 4 - Table with results from RCSIND
- 5 - Table with results from STOLWK (any)
- 6 - Table with totals from all routines
- 7 - Table for JETIND

Choose an option

1

OUTPUT DATA TO TABLE FORMAT

- 0 - Quit table generation section
- 1 - Output table with all hover results (dL/dT vs Height)
- 2 - Tables with results from STOLSF (any)
- 3 - Tables with results from STOLGV (any)
- 4 - Table with results from RCSIND
- 5 - Table with results from STOLWK (any)
- 6 - Table with totals from all routines
- 7 - Table for JETIND

Choose an option

6

Table generation for Totals

- 1 - Output table based on height
- 2 - Output table based on aircraft velocity
- 3 - Output table based on jet deflection angle
- 4 - Output table based on Aircraft Angle of Attack

Choose an option

1

V Number Velocity

- 1 --> 3.0
- 2 --> 6.5
- 3 --> 10.0
- 4 --> 13.5
- 5 --> 17.0
- 6 --> 20.5
- 7 --> 24.0
- 8 --> 27.5
- 9 --> 31.0
- 10 --> 34.5

Enter your choice for Aircraft Velocity (V Number)

4

D Number Def Angle

- 1 --> 81.0
- 2 --> 83.0
- 3 --> 85.0
- 4 --> 87.0
- 5 --> 89.0

6 --> 91.0
7 --> 93.0

Enter your choice for Deflection Angle (D Number)
1

A Number Angle of Attack
1 --> 6.0
2 --> 8.0
3 --> 10.0
4 --> 12.0

Enter your choice for Aircraft Angle of Attack (A Number)
1

OUTPUT DATA TO TABLE FORMAT

- 0 - Quit table generation section
- 1 - Output table with all hover results (dL/dT vs Height)
- 2 - Tables with results from STOLSF (any)
- 3 - Tables with results from STOLGV (any)
- 4 - Table with results from RCSIND
- 5 - Table with results from STOLWK (any)
- 6 - Table with totals from all routines
- 7 - Table for JETIND

Choose an option
7

A Number Angle of Attack
1 --> 6.0
2 --> 8.0
3 --> 10.0
4 --> 12.0

Enter your choice for Aircraft Velocity (V Number)
1

OUTPUT DATA TO TABLE FORMAT

- 0 - Quit table generation section
- 1 - Output table with all hover results (dL/dT vs Height)
- 2 - Tables with results from STOLSF (any)
- 3 - Tables with results from STOLGV (any)
- 4 - Table with results from RCSIND
- 5 - Table with results from STOLWK (any)
- 6 - Table with totals from all routines
- 7 - Table for JETIND

Choose an option

0

Output

The actual output from PIE is in the form of tables from which graphs can be created or look-ups can be performed.

The following tables were created from the menu choices shown in the previous section. The tables are actual output from PIE except for elimination of "zero" data sets to allow the tables to fit on the pages.

Tables 4 and 5 are basic tables used to create graphs using a Macintosh application called KaleidaGraph or a graphing application for the Silicon Graphics Iris. These two tables follow the format which is described next. The first row contains a number which the Iris application uses to distinguish between different types of input (the number is not used by KaleidaGraph.) The second row contains the title of the table. This title is a combination of the CONFIG variable and a description of the table type. The third and fourth rows contain the titles for the X and Y axis respectively. The fifth column contains the number of data sets (n) and the following (n) rows contain the labels for those data sets. The next row, following the labels, contains the number of data points. After the above headers, all the data sets are listed with the X-axis value followed by each data set value on the same row.

Table 4. Actual Output from PIE for Hover Lift Increments

1						
YAV-8B Harrier	, Hover Ground Effect					
Height						
dL/dT						
6						
OG						
SUCKDOWN						
FOUNTAIN						
LIDS						
TOTAL						
Zero						
25						
1.0000	-0.01142	-0.47852	0.42695	0.06524	0.00225	0.00000
2.0000	-0.01142	-0.21777	0.21192	0.06476	0.04750	0.00000
3.0000	-0.01142	-0.13432	0.14068	0.06449	0.05943	0.00000
4.0000	-0.01142	-0.09401	0.10519	0.06429	0.06406	0.00000
5.0000	-0.01142	-0.07056	0.08395	0.05324	0.05522	0.00000
6.0000	-0.01142	-0.05541	0.06983	0.04428	0.04729	0.00000
7.0000	-0.01142	-0.04491	0.05975	0.03790	0.04132	0.00000
8.0000	-0.01142	-0.03729	0.05075	0.03218	0.03423	0.00000
9.0000	-0.01142	-0.03154	0.04132	0.02620	0.02456	0.00000
10.0000	-0.01142	-0.02709	0.03188	0.02022	0.01360	0.00000
11.0000	-0.01142	-0.02356	0.02245	0.01424	0.00171	0.00000
12.0000	-0.01142	-0.02071	0.01302	0.00826	-0.01085	0.00000
13.0000	-0.01142	-0.01836	0.00359	0.00228	-0.02392	0.00000
14.0000	-0.01142	-0.01641	0.00225	0.00143	-0.02415	0.00000
15.0000	-0.01142	-0.01477	0.00210	0.00133	-0.02276	0.00000
16.0000	-0.01142	-0.01337	0.00197	0.00125	-0.02157	0.00000
17.0000	-0.01142	-0.01217	0.00185	0.00118	-0.02056	0.00000
18.0000	-0.01142	-0.01113	0.00175	0.00111	-0.01969	0.00000
19.0000	-0.01142	-0.01023	0.00166	0.00105	-0.01893	0.00000
20.0000	-0.01142	-0.00943	0.00157	0.00100	-0.01828	0.00000
21.0000	-0.01142	-0.00873	0.00150	0.00095	-0.01770	0.00000
22.0000	-0.01142	-0.00811	0.00143	0.00091	-0.01718	0.00000
23.0000	-0.01142	-0.00755	0.00137	0.00087	-0.01673	0.00000
24.0000	-0.01142	-0.00705	0.00131	0.00083	-0.01632	0.00000
25.0000	-0.01142	-0.00660	0.00126	0.00080	-0.01596	0.00000

Table 5. Actual Output from PIE for Total Lift Increments

1 YAV-8B Harrier		TOTALS	VO14.	DFL81.	AOA 6.	
Height						
dL/dT						
6						
OGE						
SF						
G-VORTEX						
JET WAKE						
RCS						
TOTAL						
25						
1.0000	-0.01142	0.01334	-0.05691	0.00360	0.00000	-0.05139
2.0000	-0.01142	0.05747	-0.02746	0.00016	0.00000	0.01875
3.0000	-0.01142	0.06912	-0.01828	-0.00050	0.00000	0.03892
4.0000	-0.01142	0.07362	-0.01372	-0.00075	0.00000	0.04773
5.0000	-0.01142	0.06500	-0.01097	-0.00087	0.00000	0.04174
6.0000	-0.01142	0.05727	-0.00911	-0.00095	0.00000	0.03579
7.0000	-0.01142	0.05145	-0.00776	-0.00099	0.00000	0.03128
8.0000	-0.01142	0.04452	-0.00675	-0.00102	0.00000	0.02533
9.0000	-0.01142	0.03510	-0.00596	-0.00104	0.00000	0.01668
10.0000	-0.01142	0.02440	-0.00532	-0.00106	0.00000	0.00660
11.0000	-0.01142	0.01281	-0.00480	-0.00106	0.00000	-0.00447
12.0000	-0.01142	0.00056	-0.00436	-0.00108	0.00000	-0.01630
13.0000	-0.01142	-0.01219	-0.00399	-0.00108	0.00000	-0.02868
14.0000	-0.01142	-0.01242	-0.00367	-0.00109	0.00000	-0.02860
15.0000	-0.01142	-0.01106	-0.00340	-0.00109	0.00000	-0.02697
16.0000	-0.01142	-0.00991	-0.00316	-0.00110	0.00000	-0.02559
17.0000	-0.01142	-0.00892	-0.00295	-0.00111	0.00000	-0.02440
18.0000	-0.01142	-0.00807	-0.00276	-0.00111	0.00000	-0.02336
19.0000	-0.01142	-0.00734	-0.00260	-0.00111	0.00000	-0.02247
20.0000	-0.01142	-0.00669	-0.00244	-0.00111	0.00000	-0.02166
21.0000	-0.01142	-0.00613	-0.00231	-0.00111	0.00000	-0.02097
22.0000	-0.01142	-0.00563	-0.00219	-0.00111	0.00000	-0.02035
23.0000	-0.01142	-0.00519	-0.00208	-0.00111	0.00000	-0.01980
24.0000	-0.01142	-0.00479	-0.00197	-0.00111	0.00000	-0.01929
25.0000	-0.01142	-0.00443	-0.00189	-0.00111	0.00000	-0.01885

Table 6. is a lookup table which is compatible with ACSYNT. (This table is too wide to fit on the page so each row is wrapped to the next row and then indented.) The first row contains three values. The first of these values is the number of heights, the second value is the number of velocities and the third number is the number of nozzle deflection angles. The second row contains the nozzle deflection angle and a listing of all the velocities. The next row contains

a height and the total lift increment for each velocity. This is repeated until all heights have been used then the nozzle deflection angle is changed and a new table is created. This happens until all nozzle deflection angles have been used. (A portion of the table is presented due to the length of the whole table.)

Table 6. Actual Output from PIE for Total Lift Increments
for ACSYNT

25	10	7			
81.000	3.0000	6.5000	10.000	13.500	17.000
	20.500	24.000	27.500	31.000	34.500
1.0000	-.14986E-01	-.28966E-01	-.40856E-01	-.51386E-01	-.60806E-01
	-.69276E-01	-.76886E-01	-.72290E-01	-.78350E-01	-.81280E-01
2.0000	0.37814E-01	0.30784E-01	0.24534E-01	0.18754E-01	0.13274E-01
	0.80638E-02	0.30438E-02	0.96000E-02	0.49100E-02	0.96300E-02
3.0000	0.52194E-01	0.47404E-01	0.43044E-01	0.38924E-01	0.34914E-01
	0.30984E-01	0.27134E-01	0.34710E-01	0.30890E-01	0.36540E-01
4.0000	0.58044E-01	0.54384E-01	0.51004E-01	0.47734E-01	0.44494E-01
	0.41284E-01	0.38064E-01	0.46230E-01	0.42940E-01	0.48370E-01
5.0000	0.50254E-01	0.47264E-01	0.44474E-01	0.41744E-01	0.39004E-01
	0.36244E-01	0.33434E-01	0.41990E-01	0.39070E-01	0.44000E-01
6.0000	0.43074E-01	0.40564E-01	0.38174E-01	0.35794E-01	0.33414E-01
	0.30954E-01	0.28424E-01	0.37250E-01	0.34570E-01	0.39020E-01
7.0000	0.37664E-01	0.35474E-01	0.33384E-01	0.31284E-01	0.29124E-01
	0.26904E-01	0.24614E-01	0.33620E-01	0.31130E-01	0.35120E-01
8.0000	0.31024E-01	0.29104E-01	0.27234E-01	0.25334E-01	0.23374E-01
	0.21334E-01	0.19184E-01	0.28370E-01	0.26020E-01	0.29580E-01
9.0000	0.21834E-01	0.20124E-01	0.18424E-01	0.16684E-01	0.14874E-01
	0.12964E-01	0.10944E-01	0.20270E-01	0.18030E-01	0.21240E-01
10.000	0.11324E-01	0.97638E-02	0.82238E-02	0.66038E-02	0.49138E-02
	0.31238E-02	0.12138E-02	0.10620E-01	0.84800E-02	0.11370E-01
11.000	-.10625E-03	-.15262E-02	-.29562E-02	-.44663E-02	-.60662E-02
	-.77662E-02	-.95862E-02	-.10000E-03	-.21500E-02	0.47000E-03
12.000	-.12226E-01	-.13536E-01	-.14876E-01	-.16296E-01	-.17806E-01
	-.19426E-01	-.21166E-01	-.11610E-01	-.13600E-01	-.11310E-01
13.000	-.24866E-01	-.26086E-01	-.27346E-01	-.28676E-01	-.30126E-01
	-.31686E-01	-.33356E-01	-.23740E-01	-.25670E-01	-.23040E-01
14.000	-.24996E-01	-.26126E-01	-.27326E-01	-.28596E-01	-.29986E-01
	-.31476E-01	-.33116E-01	-.23450E-01	-.25030E-01	-.22490E-01
15.000	-.23556E-01	-.24626E-01	-.25746E-01	-.26966E-01	-.28306E-01
	-.29756E-01	-.31316E-01	-.21560E-01	-.22820E-01	-.20510E-01
16.000	-.22336E-01	-.23346E-01	-.24416E-01	-.25586E-01	-.26866E-01
	-.28286E-01	-.29816E-01	-.19700E-01	-.20970E-01	-.18880E-01
17.000	-.21286E-01	-.22236E-01	-.23256E-01	-.24396E-01	-.25626E-01
	-.26996E-01	-.28416E-01	-.18130E-01	-.19410E-01	-.17310E-01
18.000	-.20376E-01	-.21276E-01	-.22266E-01	-.23356E-01	-.24556E-01
	-.25896E-01	-.27076E-01	-.16790E-01	-.17990E-01	-.15940E-01
19.000	-.19596E-01	-.20456E-01	-.21406E-01	-.22466E-01	-.23636E-01

Table 6. (Continued)

	-.24956E-01	-.25916E-01	-.15660E-01	-.16680E-01	-.14810E-01
20.000	-.18896E-01	-.19726E-01	-.20636E-01	-.21656E-01	-.22806E-01
	-.23936E-01	-.24916E-01	-.14500E-01	-.15560E-01	-.13840E-01
21.000	-.18296E-01	-.19096E-01	-.19976E-01	-.20966E-01	-.22096E-01
	-.23056E-01	-.24036E-01	-.13520E-01	-.14610E-01	-.13040E-01
22.000	-.17766E-01	-.18526E-01	-.19376E-01	-.20346E-01	-.21486E-01
	-.22296E-01	-.23156E-01	-.12690E-01	-.13790E-01	-.12340E-01
23.000	-.17286E-01	-.18026E-01	-.18846E-01	-.19796E-01	-.20796E-01
	-.21616E-01	-.22416E-01	-.11950E-01	-.13070E-01	-.11740E-01
24.000	-.16856E-01	-.17566E-01	-.18376E-01	-.19286E-01	-.20176E-01
	-.20966E-01	-.21756E-01	-.11310E-01	-.12460E-01	-.11220E-01
25.000	-.16476E-01	-.17156E-01	-.17936E-01	-.18846E-01	-.19646E-01
	-.20356E-01	-.21176E-01	-.10730E-01	-.11900E-01	-.10760E-01
83.000	3.0000	6.5000	10.000	13.500	17.000
	20.500	24.000	27.500	31.000	34.500
1.0000	-.15476E-01	-.29926E-01	-.42256E-01	-.53096E-01	-.62806E-01
	-.71486E-01	-.79286E-01	-.74810E-01	-.80980E-01	-.84900E-01
2.0000	0.38094E-01	0.30814E-01	0.24344E-01	0.18364E-01	0.12714E-01
	0.73437E-02	0.21737E-02	0.86000E-02	0.37800E-02	0.47400E-02
3.0000	0.52674E-01	0.47704E-01	0.43194E-01	0.38924E-01	0.34794E-01
	0.30744E-01	0.26774E-01	0.34230E-01	0.30300E-01	0.32140E-01
4.0000	0.58624E-01	0.54824E-01	0.51334E-01	0.47944E-01	0.44624E-01
	0.41304E-01	0.37984E-01	0.46050E-01	0.42670E-01	0.44560E-01
5.0000	0.50774E-01	0.47694E-01	0.44814E-01	0.41974E-01	0.39144E-01
	0.36314E-01	0.33404E-01	0.41870E-01	0.38850E-01	0.40580E-01
6.0000	0.43534E-01	0.40934E-01	0.38454E-01	0.36004E-01	0.33534E-01
	0.31004E-01	0.28404E-01	0.37150E-01	0.34390E-01	0.35920E-01
7.0000	0.38074E-01	0.35814E-01	0.33654E-01	0.31464E-01	0.29254E-01
	0.26954E-01	0.24594E-01	0.33540E-01	0.30980E-01	0.32290E-01
8.0000	0.31394E-01	0.29404E-01	0.27464E-01	0.25494E-01	0.23474E-01
	0.21364E-01	0.19164E-01	0.28270E-01	0.25860E-01	0.26990E-01
9.0000	0.22104E-01	0.20344E-01	0.18574E-01	0.16774E-01	0.14904E-01
	0.12944E-01	0.10874E-01	0.20110E-01	0.17830E-01	0.18790E-01
10.000	0.11514E-01	0.99038E-02	0.82937E-02	0.66238E-02	0.48638E-02
	0.30238E-02	0.10638E-02	0.10410E-01	0.82200E-02	0.90300E-02
11.000	-.26248E-04	-.15062E-02	-.29862E-02	-.45462E-02	-.61862E-02
	-.79462E-02	-.98062E-02	-.38000E-03	-.24900E-02	-.18300E-02
12.000	-.12266E-01	-.13626E-01	-.15016E-01	-.16486E-01	-.18036E-01
	-.19716E-01	-.21506E-01	-.11990E-01	-.14050E-01	-.13500E-01
13.000	-.25026E-01	-.26286E-01	-.27596E-01	-.28976E-01	-.30456E-01
	-.32066E-01	-.33796E-01	-.24230E-01	-.26210E-01	-.25450E-01
14.000	-.25166E-01	-.26356E-01	-.27576E-01	-.28876E-01	-.30316E-01
	-.31866E-01	-.33516E-01	-.23910E-01	-.25700E-01	-.24740E-01
15.000	-.23706E-01	-.24806E-01	-.25976E-01	-.27226E-01	-.28606E-01
	-.30086E-01	-.31706E-01	-.22060E-01	-.23420E-01	-.22610E-01
16.000	-.22466E-01	-.23506E-01	-.24606E-01	-.25826E-01	-.27136E-01
	-.28586E-01	-.30166E-01	-.20210E-01	-.21520E-01	-.20840E-01
17.000	-.21406E-01	-.22386E-01	-.23456E-01	-.24616E-01	-.25886E-01
	-.27306E-01	-.28876E-01	-.18600E-01	-.19920E-01	-.19250E-01

Table 6. (Continued)

18.000	-.20486E-01	-.21426E-01	-.22436E-01	-.23566E-01	-.24806E-01
	-.26176E-01	-.27476E-01	-.17220E-01	-.18520E-01	-.17760E-01
19.000	-.19696E-01	-.20576E-01	-.21556E-01	-.22646E-01	-.23856E-01
	-.25206E-01	-.26286E-01	-.16030E-01	-.17160E-01	-.16530E-01
20.000	-.18986E-01	-.19846E-01	-.20786E-01	-.21836E-01	-.23016E-01
	-.24266E-01	-.25246E-01	-.14920E-01	-.15990E-01	-.15460E-01
21.000	-.18386E-01	-.19196E-01	-.20106E-01	-.21146E-01	-.22286E-01
	-.23366E-01	-.24376E-01	-.13900E-01	-.15010E-01	-.14570E-01
22.000	-.17856E-01	-.18636E-01	-.19506E-01	-.20526E-01	-.21656E-01
	-.22566E-01	-.23506E-01	-.13030E-01	-.14150E-01	-.13820E-01
23.000	-.17356E-01	-.18116E-01	-.18966E-01	-.19956E-01	-.21046E-01
	-.21866E-01	-.22716E-01	-.12280E-01	-.13420E-01	-.13150E-01
24.000	-.16936E-01	-.17656E-01	-.18496E-01	-.19456E-01	-.20416E-01
	-.21236E-01	-.22026E-01	-.11610E-01	-.12770E-01	-.12580E-01
25.000	-.16546E-01	-.17246E-01	-.18066E-01	-.18996E-01	-.19856E-01
	-.20616E-01	-.21436E-01	-.11020E-01	-.12200E-01	-.12070E-01

Programmer's Guide

Overall Scheme & Assumptions

The scheme for the Power Induced Effects module is as follows; 1) Obtain a complete set of variables for the configuration. 2) Isolate variables to send to lift increment routines based on configuration and planform. 3) Calculate lift increments. 4) Store results in indexed arrays and files. 5) Output configuration variables for the user. 6) Output tables for graphing purposes. Most of the empirical equations have been developed for aircraft with high disk loading jets or fans.

Equations for Subroutines

Hover

The only independent variable used in the hover calculations is height. The other independent variables are set to zero velocity, 90° nozzle deflection angle, and 0° angle of attack.

The method used in this study was developed by Kuhn (reference 3) and assumes the total induced lift increment developed on a V/STOL aircraft can be expressed as:

$$\frac{\Delta L}{T} = \frac{\Delta L_{\infty}}{T} + \frac{\Delta L_S}{T} + \frac{\Delta L_F}{T} + \frac{\Delta L_L}{T} \quad (1)$$

Suckdown

$\frac{\Delta L_{\infty}}{T}$ is the out-of-ground effect (OGE) suckdown lift increment. The equation for the OGE suckdown is:

$$\frac{\Delta L_{\infty}}{T} = -.00022 \sqrt{\frac{S}{A}} \left[\left(\frac{P_n}{P_{\infty}} \right)^{.64} \frac{\Sigma \pi d}{d_c} \right]^{1.58} \quad (2)$$

where S is the total planform area, A is the total jet exit area, P_n/P_{∞} is the nozzle pressure ratio, $\Sigma \pi d$ is the total jet perimeter, and d_c is the diameter of an equivalent single jet having area A. Experiments have shown that the OGE lift increment is reduced if the nozzles are extended below the surface of a body. The OGE lift increment for a high wing configuration is lower than the low wing configuration and the decrease is related to the square root of the wing height over nozzle diameter as shown in equation (3).

$$\left(\frac{\Delta L_{\infty}}{T}\right)_{hw} = \left(\frac{\Delta L_{\infty}}{T}\right)_b + \left[\left(\frac{\Delta L_{\infty}}{T}\right)_{wb} - \left(\frac{\Delta L_{\infty}}{T}\right)_b\right] \left[1 - .4\sqrt{\frac{\Delta h}{d_c}}\right] \quad (3)$$

where Δh indicates the wing height above the nozzles of the configuration, and the subscripts hw indicates a high wing, b indicates the body, and wb indicates the wingbody.

$\frac{\Delta L_S}{T}$ is the additional lift increment due to suckdown experienced in-ground effect (GE). It was first determined experimentally for single jet configurations and then a correction factor was developed for multiple jet configurations. The equation for in-ground effect suckdown is:

$$\frac{\Delta L_S}{T} = K_S (-0.15) \left[\frac{\frac{h}{d_c}}{\frac{\bar{D}}{d_c} - 1.0} \right]^{2.2 - .24 \left(\frac{P_n}{P} - 1 \right)} \quad (4)$$

Where in addition to the variables mentioned above, \bar{D} is the angular mean diameter of the planform, and K_S is the multiple jet correction factor which is a function of configuration geometry. This GE suckdown also changes with wing height similar to the OGE suckdown. This is because the wing does not experience the same magnitude of entrainment as the lower surface of the body.

Fountain Lift

Two methods were developed to calculate the fountain lift, one for widely spaced jets, the Basic Method, and one for closely space jets, the h' Method.

Basic Method

The Basic Method was developed for widely spaced jets with the equal thrusts. $\frac{\Delta L_F}{T}$ is the overall fountain lift increment which is a combination of the lift increment developed by the fountain arms and core.

$$\frac{\Delta L_F}{T} = \frac{\Delta L_A}{T} + \frac{\Delta L_C}{T} \quad (5)$$

Equations (6) and (7) are used to calculate the fountain arm contribution. Equation (6) is the lift increment for an individual fountain arm 'x'. Equation (7) combines the fountain arms in the configuration.

$$\frac{\Delta L_{Ax}}{T} = \frac{2(Y_x S_x')}{N(e_x S_x'')^{.835}} \left(\frac{e_x}{e_x + h} \right)^2 \frac{y_x}{\sqrt{y_x^2 + (e_x + h)^2}} \quad (6)$$

$$\frac{\Delta L_A}{T} = \frac{1}{2} \left(\frac{\Delta L_{A,1}}{T} + \frac{\Delta L_{A,2}}{T} + \dots + \frac{\Delta L_{A,N}}{T} \right) \left(.7 \sqrt{\frac{\frac{h}{d_e}}{\frac{D}{d_c} - 1}} \right) \quad (7)$$

y_x is the spanwise extent of the jet on the planform, Y_x is the maximum spanwise extent of the jet on the planform, e is half the distance between adjacent jets, h is the height of the lowest surface above the ground, S' is the actual surface area between the jets, S'' is the potential surface area between the jets, and N is the number of fountain arms. These variables can be seen in Figure #2

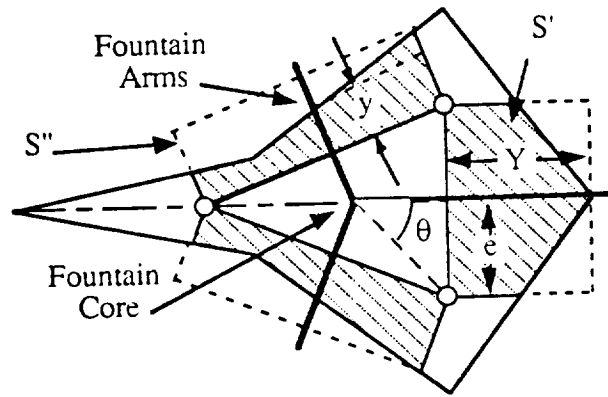


Figure #2 Fountain variables from a configuration with three jets.

The calculations for the fountain core are similar to those of the fountain arms except the contribution of each fountain arm on the fountain core is;

$$\frac{\Delta L_{Cx}}{T} = K_C \left(\frac{e_x}{e_x + h} \right)^{\lambda_C} \cos \theta_x \quad (8)$$

$$\frac{\Delta L_{C,1}}{T} + \frac{\Delta L_{C,2}}{T} + \dots + \frac{\Delta L_{C,N}}{T} \quad (9)$$

Where the constant K_C and λ_C depend on the configuration geometry and altitude, e is half the distance between adjacent jets, h is the height of lowest

surface above the ground, and θ is half the angle between adjacent jets and the center of the jet pattern.

h' Method

The h' Method was developed because configurations with closely spaced jets exhibit an abrupt increase in fountain lift at low altitudes which is not predicted with the Basic Method. Closely spaced jets have higher pressures between the jets which the Basic Method fails to predict. These higher pressures increase rapidly as the jet spacing is reduced. The ability to contain this pressure breaks down as the height is increased because the jets tend to merge into a single jet before they reach the ground.

Two different equations are used for the fountain increment. The decision for which equation to use is based on an altitude, h' .

$$\frac{\Delta L_F}{T} = K' \left(\frac{h}{d_c} \right)^{\lambda'} \quad (10)$$

$$\frac{\Delta L_F}{T} = 0.033 \left(\frac{\bar{D}w}{d_c l} \right) \left(\frac{d_c}{h} \right) \quad (11)$$

where K' and λ' are parameters calculated based on jet spacing, diameter, number, surface area, etc..., w is width of the configuration, and l is the length of the configuration. The two curves do not intersect but are joined by a straight line which is tangent to the curve of equation (10) and is projected to $\frac{\Delta L_F}{T} = 0$ at a height defined as h' . This is shown in the graph of Figure #3.

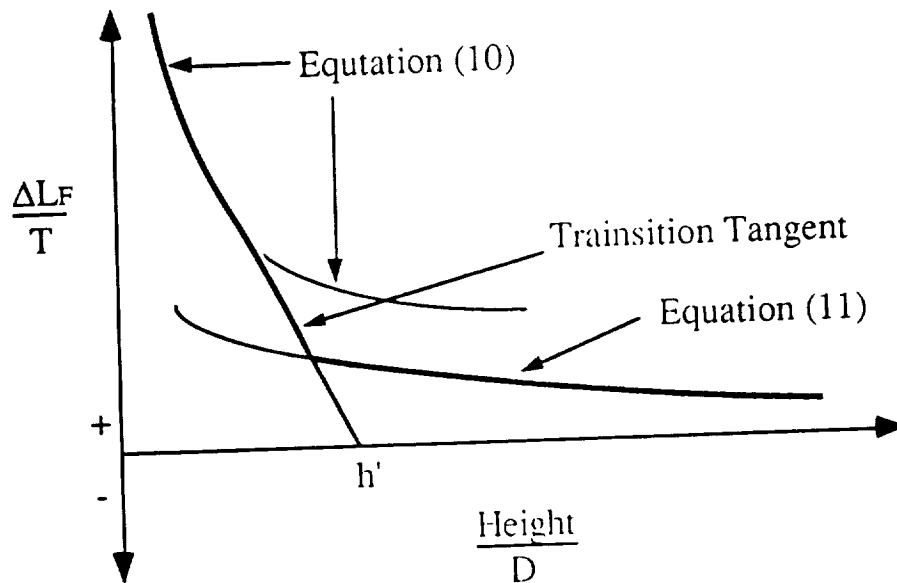


Figure #3 Typical variation of fountain lift increment using h' Method

Lift Improvement Devices

$\frac{\Delta L_L}{T}$ is the lift improvement devices (LIDs) lift increment. This lift increment is an addition to the total fountain by a percentage, K_L , of the total fountain lift. K_L can be determined using the area inclosed by the LIDs, ratio of perimeter by LIDs to the total perimeter area enclosed by LIDs, area enclosed by jet centers, length-to-width ratio of jet pattern, and altitude of aircraft. These variables can be seen in Figure #4.

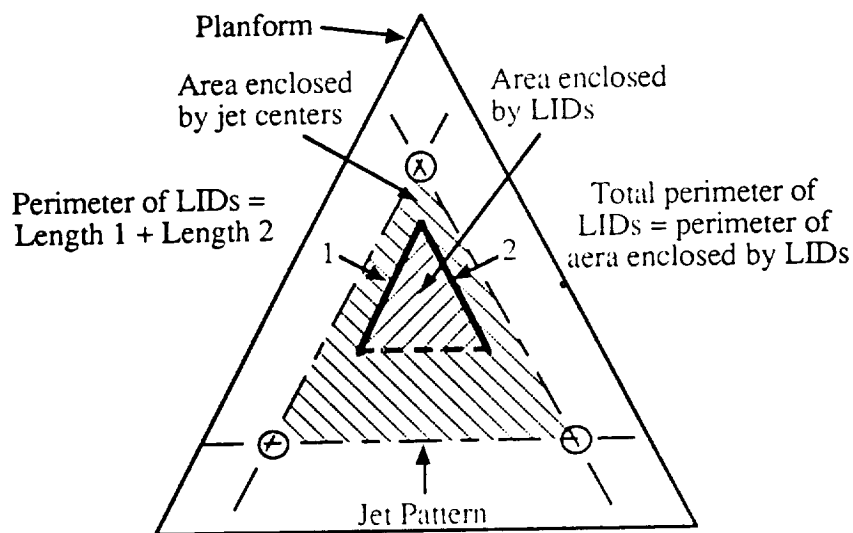


Figure #4 Lift Improvement Device definitions from a triangular configuration with three jets.

Forward Flight / Jet Deflection Angle / Angle of Attack

The calculations based on forward flight, jet deflection angle, and angle of attack use hover calculations as a basis for each lift increment calculation. The method assumes that the total lift can be expressed as:

$$\frac{\Delta L}{T} = \frac{\Delta L_S}{T} + \frac{\Delta L_F}{T} + \frac{\Delta L_{JW}}{T} + \frac{\Delta L_V}{T} \quad (12)$$

where the subscript S refers to the suckdown lift increment (no fountain), F refers to the fountain lift increment, JW refers to the jet wake lift increment, and V refers to the lift increment due to the ground vortex.

Suckdown/Fountain

The suckdown and fountain terms are calculated using a method developed in Reference 6. This method calculates a factor based on the forward extent of the wall jet and the jet spacing of the front nozzles. This factor, K_h , varies between one (the wall jet extending ahead of the configuration) and zero (the wall jet swept behind the configuration.) This factor is multiplied by the square of the sine of the jet deflection angle (δ). (Angle of attack is not used in these calculations.) This calculation produces a factor which modifies the suckdown and fountain terms for the effects of forward speed and jet deflection angle. The suckdown lift increment for forward flight and jet deflection angle is calculated as follows:

$$\frac{\Delta L_S}{T} = \left(\frac{\Delta L_S}{T} \right)_{\text{hover}} K_h \sin^2 \delta \quad (13)$$

The fountain lift increment is calculated using the fountain and LID lift increments from the hover calculations as shown in equation (14).

$$\frac{\Delta L_F}{T} = \left[\frac{\Delta L_F}{T} + \frac{\Delta L_L}{T} \right]_{\text{hover}} \sqrt{\frac{h_f}{h'}} K_h \sin^2 \delta \quad (14)$$

This lift increment calculation is very similar to equation (13) except for the $\sqrt{\frac{h_f}{h'}}$ term. This term is included due to the recommendation of Richard Kuhn. It is ratio of h' , which was described earlier and h_f , which is the maximum height to which the fountain effects are felt on the airframe.

Ground Vortex

The ground vortex term is calculated by assuming the configuration is composed of a body and a wing. The lift increment is based on the thrust, diameter and area of the front jets. There are two critical heights associated with the ground vortex lift increments; h_1 is the height at which the rate of change of lift with height changes and h_2 is the maximum height at which the ground vortex effects are felt. The equations are different for the wing and body and each of those are different for heights based on h_1 and h_2 . An example of these relations is the equation for a body at a height between h_1 and h_2 , equation (14).

$$\frac{\Delta L_V}{T} = \frac{T_f}{T} \cdot 18 \left(\frac{w}{l} \right)_f^2 \left[\left(\frac{S_B}{A_f} \right) \left(\frac{w}{l} \right)_B \right]^{.62} \sqrt{\frac{1}{V_c}} \Delta C_p [1 + \sin(\delta - 90)]^2 \left(\frac{h}{d_f} \right)^{-3.2} \quad (14)$$

The other equations are similar to equation (14) in complexity and format and they can be found in reference 6. The subscript f refers to the front jets, and B refers to the body configuration. T is the thrust, w is the width, l is the length, S and A are area, V_e is the ratio of the aircraft dynamic pressure to the jet dynamic pressure, ΔC_p is a factor based on V_e and height, and δ is the jet deflection angle.

Jet Wake

The lift increment due to the jet wake is the sum of the increments due to each jet on the wing and body. This is shown in equation (15) where B refers to the body planform, W refers to the wing planform, f refers to the front jets and r refers to the rear jets.

$$\frac{\Delta L_{wake}}{T} = \left[\left(\frac{\Delta L_f}{T} \right) + \left(\frac{\Delta L_r}{T} \right) \right]_B + \left[\left(\frac{\Delta L_f}{T} \right) + \left(\frac{\Delta L_r}{T} \right) \right]_W \quad (15)$$

The lift increment for the wake is figured by first calculating the lift increment out-of-ground effect and then including this term in the overall lift increment in-ground effect. The OGE jet wake lift increment is calculated using equation (16), which is the equation for the lift increment on a square planform with the jet at the center, and multiplying it by adjustment factors for the planform aspect ratio, longitudinal, vertical, and lateral position of the jet, distance of the jet center from the side of the body, jet deflection angle, non-circular nozzles, jet pressure ratio, and jet flap (Reference 2 and 6).

$$\frac{\Delta L_{w,square}}{T} = \left[-3 V_c^2 \left(\sqrt{\frac{S}{A}} - 1 \right)^{.67} + 35 V_c^{5.5} \left(\sqrt{\frac{S}{A}} - 1 \right) \right] \quad (16)$$

The OGE lift increment is included in equation (17) which is the equation to calculate the jet wake lift increment for an individual jet on a body planform.

$$\frac{\Delta L_{w,x}}{T} = \frac{\Delta L_{w,OGE}}{T} \left[1 - .7 \sqrt{\left(\frac{S}{A} \right)_B \left(\frac{W}{l} \right)_B} \left(\frac{\delta}{90} \right)^{1.34} \left(\frac{w}{l} \right)_j^{-.35} \left(\frac{h}{d} \right)^{-2} \right] \quad (17)$$

Similar equations are used in calculations for the wing planform (Reference 6). The subscript j refers to the jets

If the jets are placed near the trailing edge of the wing, there is a positive lift increment instead of a negative. This is due to the jet flap effect. In this case a positive lift increment is calculated and replaces the OGE term in equation similar to equation (17).

Reaction Control System

The effectiveness of the roll control jets near the wing tip during transition depends on the proximity of the control jets to the wing tip and the wing trailing edge, and jet pressure ratio (P_j/P_∞). The empirical equations developed for this lift increment are based on an integration of the pressure distributions on the surface areas around the nozzles.

The lift increments are assumed to be made up of two terms, one which is not affected by pressure ratios (O) and another from the effect of pressure ratios above a critical pressure ratio of 1.893 (P).

$$\frac{\Delta L_{RCS}}{T} = \left(\frac{\Delta L}{T} \right)_O + \left(\frac{\Delta L}{T} \right)_P \quad (18)$$

where:

$$\left(\frac{\Delta L}{T}\right)_O = (3V_c^3 - 2.4V_c^2) \sqrt{\frac{S}{A_j}} + .41V_c^{2.2} \left(\frac{S}{A_j}\right)^{.688} \quad (19)$$

$$\left(\frac{\Delta L}{T}\right)_P = -.017V_c \left(\frac{S}{A}\right)^{.42} \left(\frac{P_j}{P_\infty} - 1.893\right)^{.75} \quad (20)$$

The variable S is the area of the wing, A_j is the jet exit area, P_j is the jet exit pressure, and P_∞ is the atmospheric pressure. These equations reproduce wind tunnel data with reasonable accuracy up to effective velocity ratios of $V_e = .1$, planform-to-jet area ratios up to 7000, and jet pressure ratios up to 45.

The magnitude of the lift increment decreases as the jet is moved closer to the wing tip or trailing edge. This is because the total area near the jet is decreased which reduces the effect of the negative pressures caused by the jet. Equation (21), (22), and (23) show the jet location adjustment factors and how they affect the total RCS lift increment.

$$K_b = .25 + .2 \left(\frac{y}{d_c}\right)^{.58} \quad (21)$$

$$K_c = .25 + .06 \frac{x}{d_c} \quad (22)$$

$$\frac{\Delta L_{RCS}}{T} = \left[\left(\frac{\Delta L}{T}\right)_O + \left(\frac{\Delta L}{T}\right)_P \right] K_b K_c \quad (23)$$

K_b is the adjustment factor for the proximity of the jet to the wing tip, y is the distance from the trailing edge to the jet, K_c is the adjustment factor for the proximity of the jet to the wing trailing edge, and x is the distance from the wing trailing edge to the jet.

Routine Descriptions

All of the following routines make up PIE. They consist of nine groups; Control, Preliminary, Hover, Suckdown/Fountain, Ground Vortex, Jet Wake, Reaction Control System, Output and Miscellaneous. Almost all variables are transferred between routines using multiple, named common blocks. Each common block contains variables which can be grouped logically. The common blocks are; CONPIE, FIGPIE, HOVPIE, PIEFLAGS, PIEGV, PIERCS, PIERROR, PIESF, PIEWK, POLYGON, RESULTS.

Control

The Power Induced Effects module (PIE) is controlled by the control program, COPPIE, which coordinates the execution of PIE. COPPIE first calls

INPIE to determine all the user defined variables. Output and scratch files are opened, planform modifications are made, and final calculations for variables are finished. Next, all lift increments are calculated within a nested do-loop structure. Each increment routine is positioned inside the do-loops according to the independent variables used for each increment. After all increments are calculated, the output routine is called and then all necessary files are closed.

COPPIE contains the ACSYNT control variable, ICALC. Currently this variable is set to the appropriate value as execution proceeds through the routine. When PIE is to be incorporated with ACSYNT, the ICALC definitions need to be removed so PIE can be controlled by ACSYNT

Preliminary Routines

INPIE

This routines first defines the namelist, PIENAM, and then sets all the default values for each variable. INPIE then reads in the user defined variables from unit 9. Next, INPIE opens the file which contains the planform coordinates, reads the header and reads the appropriate planform coordinates. This is repeated until all coordinates from all planforms have been read from the file. If only a WINGBODY planform is entered then the flag WBFLAG is set to true.

PLANMO

PLANMO's first task is to determine the maximum and minimum points of the planform and then determine .1% of the difference between the two to add that to any point which has the same X-coordinate as the previous point. (This is done because the slope of a line between two points with the same X-coordinate is infinity.) At all time PLANMO checks which planform is being used.

The second task is to combine the individual body and wing planforms into one planform, the wingbody. As a by-product of this combination, the center section planform is determined. Both of these planforms are found by determining where the body and wing planforms cross and saving those points and combining the appropriate planforms together.

The last task for PLANMO is to move the most forward part of the planforms (generally the aircraft nose) to the origin of the coordinate system. Then move all other positioned items the same amount. These items include; body planform, wing planform, wingbody planform, center section planform, nozzles, center of gravity, center of area, trailing edge, RCS nozzles.

FINVAR

The purpose of FINVAR is to calculated all variables which have not been defined. Variables which have not been defined are those variables which have been set to the "undefined" flag value of 9999.0. All variables are checked with this flag before they are calculated except those calculated from the

SDAREA routine. Most calculations in this routine are very simple and are too numerous for complete description in this manual. (See code for individual calculations.)

SDAREA

The SDAREA routine was written to determine all variables which are used in fountain calculations. These include half the angle between adjacent jets and the center of the jet pattern (θ), half the distance between adjacent jets (e), maximum spanwise extent of jet on planform (Y), spanwise extent of the jet on the planform (y), actual surface area between the jets (S'), potential surface area between the jets (S'').

The routine first determines angles and defines important lines on the configuration. Then the routine divides the planform into thin slices which are summed over the entire planform to determine the areas for the configuration. During this summation, areas are calculated for individual regions of the planform the spanwise extents of the planforms are determined.

This routine was originally written as a stand-alone application. It was modified to allow it to be used as a subroutine. The routine returns the values to PIE by determining the nozzle configuration and assigning calculated values to the appropriate variables used in PIE.

DIABAR

This routine calculates the angular mean diameter of a planform which is the diameter of a circle which has the same circumference as the given planform. The routine divides the planform into arcs from some point in the interior of the planform (center of the jet pattern.) The lengths of these arcs are summed and then the angular mean diameter is determined from the total circumference of the planform.

This routine also has the capability of calculating areas of the planform in front and behind the interior point. There is also extensive user prompts, and output sections which are not used by PIE. These sections can be accessed by recoding PIE so IFLAG is set to one instead of zero when DIABAR is called.

Hover

HCALL

HCALL isolates the local variables within HOV_GE from the global configuration variables. The reason for this is HOV_GE modifies some of its local configuration variables and it is desired to keep the original configuration variables from changing.

HOV_GE

The HOV_GE routine is the lift increment calculation routine for configurations in hover. This routine receives its input through the HOVPIE and RESPIE common block and outputs its results to the RESPIE common block. The routine is based on the original BASIC program developed by Richard Kuhn. The lift increments which it calculates are out-of-ground effect suckdown, in-ground effect suckdown, fountain effect, and LID effect.

Suckdown/Fountain**SFCALL**

SFCALL isolates the local variables within STOLSF from the global configuration variables. It also chooses particular variables to pass to STOLSF based on the configuration.

STOLSF

The STOLSF routine calculates the suckdown/fountain lift increment as it varies with height, forward velocity and nozzle deflection angle. The routine receives input through the PIESF and RESPIE common block, makes the calculations, and then outputs the results to direct access scratch files which are used in the output routines. The results are stored in the scratch files using an equation which assign a unique record number to each result. The record numbers are calculated with the same indices used when accessing arrays.

Ground Vortex**GVCALL**

GVCALL isolates the local variables within STOLGV from the global configuration variables. This control routine is more detailed than the previously mentioned control routines. GVCALL determines which planform(s) is being used, assigns variables based on the planform, calls STOLGV routine, and then output the results to similar scratch files mentioned in STOLSF.

STOLGV

The STOLGV routine is more simplified than previous increment routines. This routine almost independent of configuration. The routine receives input through PIEGV and RESPIE common blocks, uses a lift increment equation based on the heights of the configurations and passes the results back to GVCALL through the PIEGV common block.

Jet Wake

WKCALL

WKCALL isolates the local variables within JIEPIE and STOLWK from the global configuration variables. This routine is the most detailed control routine. It first determines which planform is being used and then using that planform, makes calculations and routine calls based on the front nozzles and then based on the rear nozzles. This is repeated until all necessary planforms have been used in the calculations. All results are stored using similar scratch files mentioned in STOLSF.

JIEPIE

The purpose of JIEPIE is to calculate the change in the lift caused by the jet wake induced effects on a flat plate. JIEPIE accounts for aspect ratio, longitudinal, vertical, and lateral position of the jets, deflection angle, distance of nozzles from body, noncircular or closely spaced jets, jet-flap effect, and jet pressure ratio. The routine receives input through PIEWK and RESPIE common blocks and it passes the results back to WKCALL through the PIEWK common block.

STOLWK

The purpose of STOLWK is to calculate the lift increment due to the jet wake system. This is calculated by checking the aspect ratio of the planform and the proximity of the nozzles to the wing trailing edge. All variables are passed to STOLWK with the PIEWK and RESPIE common blocks and its results are returned to WKCALL with the PIEWK common block.

Reaction Control System

RCSCAL

RCSCAL is executed only once for each velocity because the RCS lift increment does not vary with height, nozzle deflection angle or angle of attack. The routine executes the lift increment routine only when the user has set FLGRCS to TRUE, otherwise the routine sets the RCS increment to zero.

RCSIND

The purpose of RCSIND is to make all necessary configuration calculation and lift increment calculations. All variables are passed to RCSIND with the PIERCS and RESPIE common blocks and its results are stored in the RESPIE common block using an array.

Output**PIEP**

PIEP outputs all results either to the screen or to files for later use. All variables are passed to this routine using the FIGPIE, PIEFLAG, RESPIE, and PIERROR common blocks. The first task of PIEP is output to the screen of all configuration variables in an easy to read format. This is immediately followed by a menu system from which many different tables can be produced. All results which are stored as arrays can be recalled using arrays. The results which were stored using scratch files need the record number equations to recall the specific results from the scratch files.

Most results were stored as components of the lift increments. The suckdown/fountain lift increments are stored as as suckdown component and also as a fountain component. The ground vortex and jet wake increments are stored as wing and as body components. These components must be combined when any total increments are requested.

JIETAB

JIETAB is an example table output routine. This routines outputs a table which is compatible with ACSYNT. The only variable passed as a parameter is the angle of attack number for which the table is to be produced. All other variables are passed to JIETAB using the RESPIE common block.

Miscellaneous**CENTAR**

The CENTAR routine calculates the center of area, area, or area in front of a point for a give planform depending on the value of the variable MODE. Important variables are passed as parameters.

CROSSIN

The CROSSIN routine calculates the intersection of two lines, with each line defined by two points (X,Y). The end points of each line lie on the same X-coordinate. All variables are passed using parameters.

DIST

The DIST routine calculates the distance between two points on the Cartesian plane using the distance formula. These points are passed as parameters to the routine.

INTEGRA

The INTEGRA routine calculates the areas of rectangles which are 'dx' wide and bounded by 1) two values defined by 'a' and 'b' and 2) a value defined by

'c' and zero. This routine is used when calculating areas in the SDAREA routine. All variables are passed using parameters.

LINECRO

The LINECRO routine calculates the intersection of two lines, each defined by a slope and Y-intercept. All variables are passed using parameters.

LINTERP

The LINTERP routine linear interpolates between two given points using a value on the X-axis. All variables are passed using parameters.

PERPDIS

The PERPDIS routine calculates the perpendicular distance between a point and a line. The point is defined by its X- and Y-coordinate, the line is defined by its slope and y-intercept. All variables are passed using parameters.

POLYAR

The POLYAR routine calculates the area of any polygon with up to 500 points. the points are passed to the routine using the MISC common block.

RTOTAL

The RTOTAL routine sums the different lift increments to the total lift increment. This is done given an indexed notation (similar to array notation).

SSEN

The SSEN routine is the Start, Step, End, Number calculator. Given any three values for START, STEP, END, and NUMBER the routine will calculate the undefined value. If less than three of the variables are given then the routine uses default values to calculate any variables not defined. Variables are undefined when their values are set to the flag value of 9999.0. All variables are passed using parameters.

References

1. Aircraft Synthesis Program (ACSYNT), National Aeronautics and Space Administration, Ames Research Center, Moffett Field, California
2. Henderson, C., Clark, J., Walters, M. "V/STOL Aerodynamics and Stability & Control Manual" Naval Air Development Center, Pennsylvania, January 15, 1980. NADC-80017-60.
3. Kuhn, R.E. "An Engineering Method of Estimating the Induced Lift on V/STOL Aircraft Hovering In and Out of Ground Effect." V/STOL Consultant. NADC-80246-60. pp. January, 1981.
4. Kuhn, R.E. "HOVER-GE", "JET-IND2", "RCS-IND", "STOL-GE2", "STOL-GV2", "STOL-SF3", "STOL-WK3" (BASIC computer programs) V/STOL Consultant. 1989-1990.
6. Stewart, V.R. and Kuhn, R.E. "A Method for Prediction of the Aerodynamic Stability and Control Parameters of STOL Aircraft Configurations." North American Aircraft Operations. Rockwell International Corporation. AFWAL-TR-87-3019. Volume II & III. June 1987.

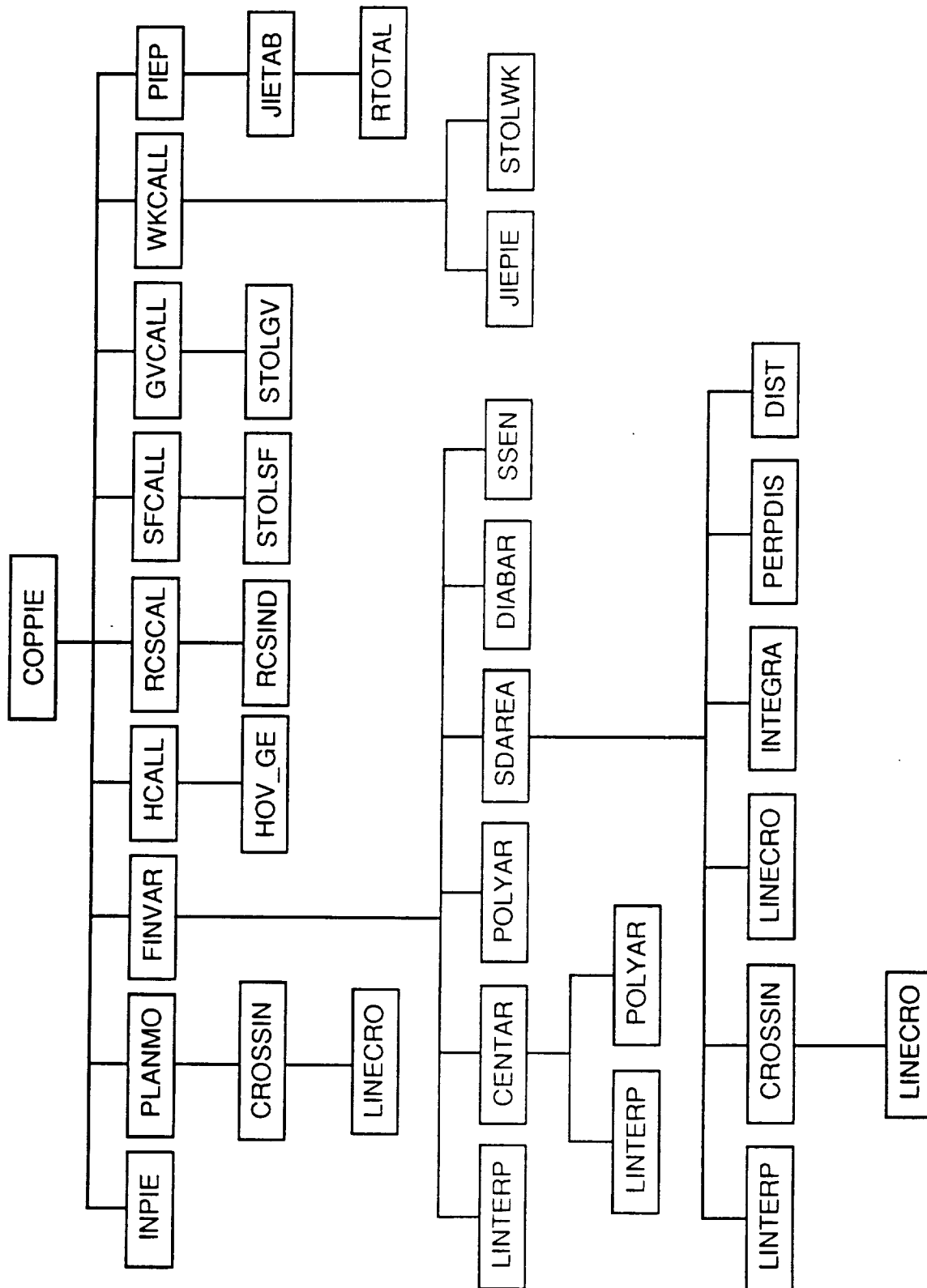
Appendices

Routine Summary		
-----------------	--	--

Routine	Acronym	Purpose
COPPIE	Control Program for the Power Induced Effects module	Controls execution and coordination of Power Induced Effects Module
INPIE	INput for Power Induced Effects module	Sets defaults for variables and reads user defined input file
PLANMO	PLANform MOdifications	Modifies planforms for consistency and integrity
FINVAR	FINish Variables	Calculates all variables which have not been defined by the user or set to a default value
SDAREA	SuckDown Area	Calculates areas necessary to make calculations for the fountain effect
DIABAR	DIAMeter BAR (\bar{D}) - compiler would not accept DBAR	Calculates angular mean diameter of a planform
HCALL	Hover CALLing routine	Isolates and controls execution of HOV_GE
HOV_GE	Hover with Ground Effects	Calculates hover lift increments for OGE and GE suckdown, and fountain effects
SFCALL	Suckdown/Fountain CALLing routine	Isolates and controls execution of STOLSF
STOLSF	STOL operation - estimation of Suckdown and Fountain effects	Calculates suckdown and fountain lift increments while in STOL flight
GVCALL	Ground Vortex CALLing routine	Isolates and controls execution of STOLGV
STOLGV	STOL operation - estimation of Ground Vortex effects	Calculates ground vortex lift increments while in STOL flight
WKCALL	jet WaKe CALLing routine	Isolates and controls execution of STOLWK
JIEPIE	Jet Induced Effects for PIE	Calculates change in lift caused by the jet induce effects on a flat plate
STOLWK	STOL operation - estimation of jet WaKe effects	Calculates jet wake lift increments while in STOL flight
RCSCAL	Reaction Control System CALLing routine	Isolates and controls execution of RCSIND

RCSIND	Reaction Control System INDuced effects	Calculates RCS lift increments while in forward flight
PIEP	Power Induced Effects Printing routine	Printing routine for PIE
JIETAB	Jet Induced Effects TABLE generation routine	Creates output file which is used by ACSYNT
CENTAR	CENTER of AREa routine	Calculates the center of area, area and area in front of a point for a given planform
CROSSIN	CROSSING routine	Calculates the intersection of two lines, with each line defined by two points
DIST	DISTance function	Calculates the distance between two points
INTEGRA	INTEGRation routine	Calculates the areas of thin rectangles
LINECRO	LINE CROSSing routine	Calculates the intersection of two lines, each defined by a slope and Y- intercept
LINTERP	Linear INTERPolation	Linear interpolates between two X and Y values give an intermediate X value
PERPDIS	PERPendicular DISTance routine	Calculates the perpendicular distance between a point and a line
POLYAR	POLYgon AREa routine	Calculates the area of any polygon
RTOTAL	Read TOTAL increments routine	Sums the total lift increment given array type indices
SSEN	Start, Step, End, Number Calculator	Calculates start, stop, end, and number variables which have not been defined

PIE Hierarchy Diagram



Cross Reference - Common Blocks

[illegible]

Cross Reference - Calls To

Module	Makes calls to:
COPPIE	INPIE, PLANMO, FINVAR, HCALL, RCSCAL, SFCALL, WKCALL, GVCALL, PIEP
INPIE	NONE
PLANMO	CROSSIN
FINVAR	LINTERP, CENTAR, POLYAR, SDAREA, DIABAR, SSEN
SDAREA	LINTERP, CROSSIN, LINECRO, INTEGRA, PERPDIS, DIST (FUNCTION)
DIABAR	NONE
HCALL	HOV_GE
HOV_GE	NONE
SFCALL	STOLSF
STOLSF	NONE
GVCALL	STOLGV
STOLGV	NONE
WKCALL	JIEPIE, STOLWK
JIEPIE	NONE
STOLWK	NONE
RCSCAL	RCSIND
RCSIND	NONE
PIEP	JIETAB
JIETAB	RTOTAL
CENTAR	LINTERP, POLYAR
CROSSIN	LINECRO
DIST	NONE
INTEGRA	NONE
LINECRO	NONE
LINTERP	NONE
PERPDIS	NONE
POLYAR	NONE
RTOTAL	NONE
SSEN	NONE

Cross Reference - Calls By

Routine	Called by:
COPPIE	NONE
INPIE	COPPIE
PLANMO	COPPIE
FINVAR	COPPIE
SDAREA	FINVAR
DIABAR	FINVAR
HCALL	COPPIE
HOV_GE	HCALL
SFCALL	COPPIE
STOLSF	SFCALL
GVCALL	COPPIE
STOLGV	GVCALL
WKCALL	COPPIE
JIEPIE	WKCALL
STOLWK	WKCALL
RCSCAL	COPPIE
RCSIND	RCSCAL
PIEP	COPPIE
JIETAB	PIEP
CENTAR	FINVAR
CROSSIN	PLANMO, SDAREA
DIST	SDAREA
INTEGRA	SDAREA
LINECRO	SDAREA, CROSSIN
LINTERP	FINVAR, SDAREA, CENTAR
PERPDIS	SDAREA
POLYAR	FINVAR, CENTAR
RTOTAL	JIETAB
SSEN	FINVAR